ENHANCING SAFETY FOR AUTONOMOUS SYSTEMS VIA REACHABILITY AND CONTROL BARRIER FUNCTIONS

by

Jason King Ching Lo

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Aeronautics and Astronautics

School of Aeronautics and Astronautics

West Lafayette, Indiana

May 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Shaoshuai Mou, Chair

School of Aeronautics and Astronautics

Dr. Inseok Hwang

School of Aeronautics and Astronautics

Dr. DengFeng Sun

School of Aeronautics and Astronautics

Approved by:

Dr. Gregory Blaisdell

Head of the School Graduate Program

To my mother, Louisa, for all the unconditional love and support she has provided.

ACKNOWLEDGMENTS

First, I would like to acknowledge my advisor, Dr. Shaoshuai Mou, for all the support and mentorship he has provided throughout my undergraduate and graduate career. A special thanks to Jessica Sun and Zihao Liang for the help and emotional support you have given. I also want to thank Wanxin Jin for filling in the gaps in my understanding on control theories as well as providing long inspiring discussions. None of this would have happened without you all. I want to thank God for his guidance and the abilities he has equipped me. Last but not least, I would like to acknowledge myself for always trying my best and never giving up.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

In this thesis, we explore different methods to enhance the safety and robustness for autonomous systems. We achieve this goal using concepts and tools from reachability analysis and control barrier functions. We first take on a multi-player reach-avoid game that involves two teams of players with competing objectives, namely the attackers and the defenders. We analyze the problem and solve the game from the attackers' perspectives via a moving horizon approach. The resulting solution provides a safety guarantee that allows attackers to reach their goals while avoiding all defenders.

Next, we approach the problem of target re-association after long-term occlusion using concepts from reachability as well as Bayesian inference. Here, we set out to find the probability identity matrix that associates the identities of targets before and after an occlusion. The solution of this problem can be used in conjunction with existing state-of-the-art trackers to enhance their robustness.

Finally, we turn our attention to a different method for providing safety guarantees, namely control barrier functions. Since the existence of a control barrier function implies the safety of a control system, we propose a framework to learn such function from a given user-specified safety requirement. The learned CBF can be applied on top of an existing nominal controller to provide safety guarantees for systems.

1. INTRODUCTION

In the past, performance has been the main emphasis when designing a controller and it can be argued that system safety did not receive as much attention as it deserves. With autonomous systems gaining ever more attention and acceptance, the safety aspect of a control system has been brought to the foreground. As we achieve higher and higher levels of autonomy, we will naturally begin to contemplate about the reliability of these autonomous systems, especially when they are capable of endangering the society and mankind. By completely eliminating human intervention, the responsibility of making sure systems do not cause dangers or be in an undesirable state now falls on the shoulders of the autonomous systems themselves. Since many autonomous systems operate in safety critical environments (rocket launches, air-traffic control, etc), it is now as important to ensure system safety as, if not more, to ensure the objective is achieved efficiently.

While many systems do have basic mechanisms or algorithms to avoid danger, more often than not there are no guarantees provided to certify the safety of the system. Moreover, different kinds of uncertainty and disturbance that exist in reality might overwhelm these algorithms and we might end up in a situation where it is too late to remedy. Furthermore, for applications that involve human-robot collaboration, not only do the robots have to complete the mission efficiently, but they must also avoid hindering the human from performing their own tasks. The above issues introduce complexity and uncertainty and makes the jobs of control engineers and system designers even more difficult than they already are.

Consider self-driving cars such as a Tesla car for instance. The vehicle must possess very robust avoidance systems and can be proved to be able to avoid pedestrians on the road. Beside that, it should always navigate at a speed such that, if a reckless pedestrian runs in front of it, there is enough "reaction" time to perform avoidance maneuvers. Obviously, the vehicle must also not go too slow or else it will be deemed inefficient in terms of "goal reaching".

Looking at the literature, the notions of safety and performance have often been treated separately; controllers that are concerned with one often neglects the other. One of the difficulties lies in that there is usually an inevitable trade-off between safety and performance.

It is not uncommon that the two aspects even directly contradicts each other. Take driving for example: the faster you drive, the less time it takes to get to the destination, but the more accident-prone you become. Depending on the application, such trade-off between safety and optimality will be different; the safety requirement for a home-cleaning Roomba is obviously much lower than that of an auto-pilot system for a Boeing 747. This means there is no right answer for "how much" safety we want in a system. However, for safety critical systems, we do know that safety must always come before performance, that is, we are only interested in optimality once safety is guaranteed. The above discussion inspires us to investigate how one can achieve a formal certification for safety and how we can do so without steering too far away from the mission objective. In this thesis, we set out to explore the questions: 1.) How can one provide safety guarantees for autonomous systems and 2.) how does one do so in a way that minimize the impact on performance?

1.1 Motivation

To address the first question, it is important that we account for these disturbances and perhaps consider the worst-case scenario when designing systems for safety critical applications. We turn our attention to the notion of "prevention" in hope of finding ways to provide such guarantees. The intuition is: if we can understand what states can lead to danger, then by avoiding these states in the first place we can *guarantee* the system never end up in undesired situation. This idea aligns with the essence of reachability analysis and hence we will be exploring methods derived from the corresponding theoretical results. At a high level, reachability tells us how "far" the system can reach given the dynamics and feasible control. Applying this backward (computing backward reachable set) can help identify the states that will potentially lead to the danger zone, which is essential for designing preventive measures.

First, we will be analyzing safety in the context of collision avoidance — more specifically, a multi-player reach-avoid game. We believe this kind of games is worth investigating since it involve elements such as complex dynamics and worst-case disturbances. By studying this

through the lens of game theory, we hope to gain better insight to situations and applications that involve dealing with conscious enemies with different degrees of hostility.

In addition, we also demonstrate how reachability can also be used in target re-association application. We show that by understanding system limits we can increase the accuracy of prediction and hence make target association more robust. This in turns means we can retain more historical information about the target and therefore are able to perform better analysis on it.

To answer the second question, we shift the focus to another idea called control barrier function. Control barrier function is similar to reachability analysis in that it can help provide formal safety guarantees for control systems. On the other hand, while it is unclear how to incorporate performance measure in the reachability framework, this can be relatively easily done via an optimization framework and control barrier functions. However, finding or constructing such control barrier function is not a trivial task and, as far as we are concerned, there are no systematic approach to do so due to its application-specific nature. This is even harder to achieve when input/actuator constraints are involve, as will be explained in section 4. We therefore see the need to develop a learning framework for control barrier functions.

1.2 Background and Related Literature

In this section, we discuss several important concepts that will be used throughout the thesis and review the related literature. The goal here is to understand the state-of-the-art in each area as well as the advantages and disadvantages of common methods used by other researchers. This will also help elucidate our motivations for the work done in this thesis.

1.2.1 Reachability Analysis

Hamilton-Jacobi reachability analysis is an important tool for verifying different complex autonomous systems. As a formal verification method, reachability analysis provide guarantees on different aspects such as safety and performance. This kind of analysis has several advantages over method that relies on simulations, especially when we are dealing with safety critical systems. First, reachability analysis considers all possible system behaviors, which

can be difficult (and exhausting) to do via simulations. Second, even if we can simulate all sorts of behaviors, real systems often involve model uncertainties and disturbances which cannot be accounted for in simulations. Reachability, on the other hand, has been shown to successfully account for worst-case disturbance through a differential game approach. By considering the worst-case scenario, one can guarantee the safety of the system as long as the disturbance does not exceed the predetermined bound. Lastly, by computing and visualizing the forward or backward reachable sets, system designers may gain better insight to the complex system.

There are multiple approaches for computing reachable sets [1][2][3][4]. In this thesis, we will mainly be discussing reachable sets in the context of the level-set approach [1]. In reachability analysis, we are often interested finding the set of states that can lead to some target states, namely the backward reachable set. These target states can be the dangerous states (for avoidance) or goal states (for goal-reaching) depending on the context. For instance, imagine a vehicle is trying to avoid collision; the target set (of states) can be the unsafe states (collision) and the backward reachable set represents the potentially unsafe states.

It is typical in reachability analysis to formulate the system we are analyzing into a two-player differential game, where one player represents the system input and the other represents the disturbance or an opposing agent. Consider a system that involves two players and its dynamics follows the ordinary differential equation

$$\dot{x}(\tau) = f(x(\tau)), a(\tau), b(\tau)), \tau \in [t, 0], a(\tau) \in \mathcal{A}, b(\tau) \in \mathcal{B} \tag{1.1}$$

where $a(\tau)$ and $b(\tau)$ are the control inputs of Player 1 and Player 2, respectively. $x \in \mathbb{R}^n$ is the system state and $a(\cdot)$ and $b(\cdot)$ are taken from a set of measurable functions, e.g.,

$$a(\cdot) \in \mathbb{A}(t) = \{\phi : [t, 0] \to \mathcal{A} : \phi(\cdot) \text{ is measurable}\}$$
$$b(\cdot) \in \mathbb{B}(t) = \{\phi : [t, 0] \to \mathcal{B} : \phi(\cdot) \text{ is measurable}\}.$$

where $\mathcal{A} \subset \mathbb{R}^{n_u}$ and $\mathcal{B} \subset \mathbb{R}^{n_d}$ are compact sets and t is negative definite. Then let us define the solution trajectories that begins at x at time t as $\zeta(\tau; x, t, a(\cdot), b(\cdot)) : [t, 0] \to \mathbb{R}^n$. Lastly, we denote the target set of interest $\mathcal{L}_0$.

Now, recall that we want to find the backward reachable set (of states). We further assume this is some sort of avoidance application where Player 1 wish to drive the system away from the target set while Player two wants to drive the system towards the target set. The BRS is then given as

$$\mathcal{L}(t) = \{x : \exists \gamma(t) \in \Gamma(t), \forall a(\cdot) \in \mathbb{A}, \zeta(0; x, t, a(\cdot), \gamma[a](\cdot)) \in \mathcal{L}_0\} \tag{1.2}$$

where $\Gamma(\cdot)$ is the set of feasible control strategies that Player 2 can use.

We note that $\Gamma(\cdot)$ actually refers to a type of strategies known as non-anticipative strategies. This is typically defined as

$$\gamma \in \Gamma(t) \coloneqq \{\beta : \mathbb{A}(t) \to \mathbb{B}(t) | a(r) = \hat{a}(r) \forall r \in [t, \tau] \Rightarrow \beta[a](r) = \beta[\hat{a}](r) \forall r \in [t, s]\}.$$

Basically, this means Player 2 is not allowed to react to Player 1's action based on Player 1's input at future time, e.g., $a(\tau) \mathrm{if} \tau > r$. In general, such non-anticipative strategy is suitable for reachability analysis since it gives an instantaneous advantage to Player 2 (our disturbance); this allow it to react Player 1's input decision with the best "counter-measures". This aligns with the idea of accounting for worst-case scenario in reachability theory.

It is important note that to obtain the BRS described in (1.2) we must solve a "game of kind" instead of a "game of degree". A "game of kind" consist of a boolean outcome; in this case, whether the target set is reached after a given period. However, traditional differential games are solved using optimal control (OC) techniques and are considered as "game of degree". This is where the level set method comes into picture: it effectively transforms a "game of kind" into a "game of degree" so we can solve it using traditional OC methods.

Before we explain the level set approach, let us first briefly discuss how differential games are solved in the OC context.

As in most optimal control problem we begin with a cost function

$$J(x, a(\cdot), b(\cdot), t) = \int_t^0 g(x(\tau), a(\tau), b(\tau), \tau)\, d\tau + h(x(0)) \tag{1.3}$$

where $g(x)$ is the running cost and $h(x)$ is the final cost. We assume Player 1 attempts to maximize this function and Player 2 wishes to minimize it. Recall that Player 2 has a slight advantage in terms of information pattern, therefore the lower value of the game is given as

$$V(x, a(\cdot), b(\cdot), t) = \min_{\Gamma[a](\cdot)} \max_{a(\cdot)} \int_t^0 g(x(\tau), a(\tau), b(\tau), \tau)\, d\tau + h(x(0)). \tag{1.4}$$

By applying dynamic programming principle the above equation can be formulate into the Hamilton-Jacobi-Isaacs (HJI) equation below

$$0 = \frac{\partial V}{\partial t} + \max_a \min_b \left[g(x, a, b) + \frac{\partial V}{\partial x} \cdot f(x, a, b) \right], \; V(x, 0) = h(x), \tag{1.5}$$

where the second term is known as the Hamiltonian $H(t, x, \nabla x) = \max_a \min_b [g(x, a, b) + \frac{\partial V}{\partial x} \cdot f(x, a, b)]$. The optimal control input for Player 1 and 2 can simply be obtained as

$$a^*(x, t) = \arg\max_{a \in \mathcal{A}} \min_{b \in \mathcal{B}} g(x, a, b, t) + \frac{\partial V}{\partial x} \cdot f(x, a, b) \tag{1.6}$$

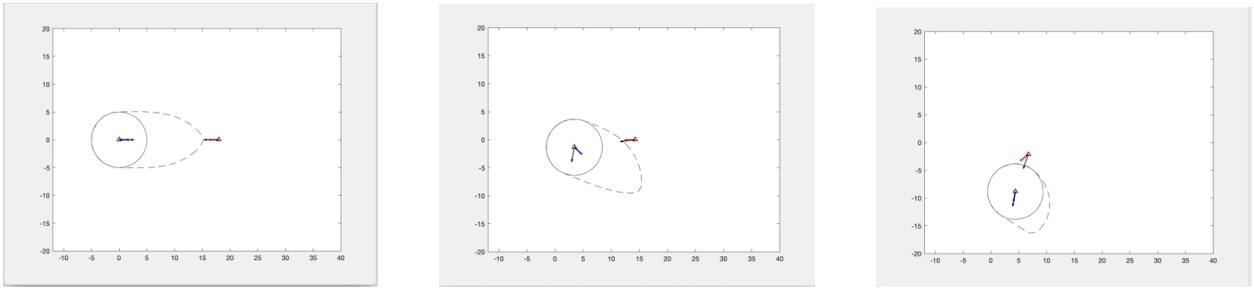
$$b^*(x, t) = \arg\min_{b \in \mathcal{B}} g(x, a, b, t) + \frac{\partial V}{\partial x} \cdot f(x, a, b) \tag{1.7}$$

Now, let us finally take a look at how the level set approach transform our "game of kind" into a "game of degree". It turns out, as described in [1], we can "encode" the boolean outcome by removing the running cost and chose the final cost "smartly". For instance in our collision avoidance example, we can choose the final cost to be the distance to the collision set *at the terminal time* T. This can be done through constructing a Lipschitz function $l(x)$ such that the target set coincides with the zero-sublevel set of this $l(x)$. Then, once we obtain the value function by solving the HJI PDE, the zero sublevel set of the value function is then the backward reachable set [5]. The BRS found represents the set of states from which the disturbance has a control sequence to drive the system in to the target set (collision). This means if we avoid this BRS in the first place, we can guarantee safety.

Apart from the backward reachable set discussed above, one can also compute the forward reachable set (FRS) using similar approach. Contrary to the BRS, the FRS represent the set of states that the system can reach from a given initial set, after a certain period. This is formally given as

$$\mathcal{F}(t) = \{\hat{x} : \exists \gamma \in \Gamma(t), \forall a(\cdot) \in \mathbb{A}, \zeta(t; x, 0, a(\cdot), \gamma[a](\cdot)) = \hat{x}, x \in \mathcal{L}_0\}, \; t > 0 \tag{1.8}$$

In [6], a toolbox is developed using the approach described above. This toolbox allows us to compute solutions for HJ-PDEs and hence obtain reachable sets for different systems. An simple simulation of the classical game of two identical vehicles is recreated using such toolbox and shown in Fig. 1.1.



(a) (b) (c)

Figure 1.1. Game of two identical vehicles: blue and red arrows indicate instantaneous control input of the evader and pursuer, respectively. Dashed line is a 2D slice of the backward reachable set.

This example is essentially a pursuer-evader game. As long as the red pursuer begins outside of the backward reachable set (dashed line), the blue evader can always find a control to avoid it. This kind of game is a special kind of zero-sum differential game [7]. Indeed, many applications that involve reachability analysis are done by posing the problem in the form of a differential game [8][9]. The application of reachability spans multiple different areas. For instance, reachability analysis has been used extensively to ensure the safety of aerial systems [10][11][12][13][14], umanned ground vehicles (UGV) [15][16][17], and human-robot interaction [18][19][19], just to name a few.

Any methods comes with a drawback and level-set approach is no exception. Although it possess many advantages as discussed above, the level set approach has a major limitation: the curse of dimensionality. Since the level-set approach requires discretizing the state-space into a grid and numerically compute the value function, the method scales exponentially with the number of dimensions. This means applying it to high-dimensional system is impractical and therefore most work in the current literature only involve applications with dimensions less than five. Intuitively, this makes sense because reachability analysis considers all possible system behaviors (although in a smart way) and solving Hamiltonian-Jacobi PDE's, so a relatively large computational cost is in a sense inevitable.

However, there are different methods developed to address this issue. One method is to decompose high dimensional systems into sub-systems to allow for fast and exact computation of BRS in lower dimensional subspaces [20][21][22]. A similar approach that simplifies system dynamics by treating certain states as disturbances make it flexible to trade of between computational cost and accuracy is proposed in [23]. Finally, many work has gone into approximating such reachable sets via different representations such as polytopic approximation [24] and ellipsoidal techniques [25].

1.2.2 Control Barrier Functions

In the previous section we talked about how reachability can be used to warrant the safety of a control system. We now turn our attention to another popular class of methods for providing safety guarantees, namely control barrier functions (CBFs). This class of methods has recently gained popularity can be attributed to the increased interest in autonomous systems, for which safety is obviously a critical aspect. In addition, the recent re-formulation of control barrier function has demonstrated its resemblance to control Lyapunov functions. In fact, many techniques involving Lyapunov functions used in stability analysis can be applied to address system safety.

In the framework of control barrier functions, the safety of a system is translated and expressed through the notion of invariant sets. At a high level, given a set of safe states and define the unsafe states to be its compliment, the safety of the system is guaranteed if we can

always find a way to remain in the safe set. Before we give the formal definition of control barrier function, we first take a brief look at what a barrier function is. Barrier function is developed based on the necessary and sufficient conditions for set invariance given by Nagumo. Nagumo's theorem states that: given a dynamical system $\dot{x} = f(x)$, where $x \in \mathbb{R}^n$, and let the safe set $\mathcal{C}$ be the zero super-level set of a smooth function $h : \mathbb{R}^n \mapsto \mathbb{R}$, e.g., $\mathcal{C} = \{x \in \mathbb{R}^n | h(x) \geq 0\}$, and $\frac{\partial h}{\partial x} \neq 0 \forall x \in \partial\mathcal{C}$, then the necessary and sufficient conditions for set invariance is:

$$\mathcal{C} \text{ is invariant} \Leftrightarrow \frac{\partial h}{\partial x} \geq 0 \, \forall x \in \partial\mathcal{C}$$

Intuitively, this condition ensures the vector field of the points on the boundary of the set are pointing "inwards", hence preventing any trajectory from "escaping" the set. In [13] the authors combined Lyapunov functions with barrier functions to create a "barrier-Lyapunov function" $\mathcal{B}(x)$, which can be used to ensure the invariance of a set through enforcing the condition $\dot{\mathcal{B}}(x) \leq 0 \; \forall x \in \mathcal{C}$.

Note that the above formulation does not involve any control inputs. In [26][27], the authors consider control systems of the form $\dot{x} = f(x) + g(x)u$, where $u \in \mathcal{U} \in \mathbb{R}^m$ and in [28] the definition of controlled barrier function was defined. Naturally, the invariant set now becomes controlled invariant set: the set of states at which there exist a controller that can control the system to stay in such set. This definition of control barrier function satisfies:

$$\exists u \;\; \text{s.t. } \dot{h}(x, u) \geq 0 \;\; \forall x \in \mathcal{C} \Rightarrow \mathcal{C} \text{ is invariant}$$

While the above definition of CBF can ensure safety through set invariance, it is overly restrictive and the corresponding conditions can be difficult to satisfy. This is because the condition essentially ensures *every* sublevel set of the safe set invariant. For this reason, a more relaxed definition of CBF is recently developed [20][21] and the new sufficient and necessary condition for set invariance is:

$$\exists u \;\; \text{s.t. } \dot{h}(x, u) \geq -\alpha(h(x)) \;\; \forall x \in \mathcal{C} \Leftrightarrow \mathcal{C} \text{ is invariant}$$

where $\alpha(\cdot)$ is an extended class $\mathcal{K}$ function. Note that this is a less restrictive condition since it allows $\dot{h}(x) \leq 0$ for $h(x) \geq 0$.

Now let us formally state the definition of control barrier functions that will be used in this thesis and state a few important related theorems. Consider a safe set $\mathcal{C} \subset \mathcal{D}$ defined to be the zero-superlevel set of a continuous differentiable function $h : \mathcal{D} \subset \mathbb{R}^n \Rightarrow \mathbb{R}$, e.g.,

$$\mathcal{C} = \{x \in \mathbb{R}^n | h(x) \geq 0\}, \tag{1.9}$$

$$\partial\mathcal{C} = \{x \in \mathbb{R}^n | h(x) = 0\}, \tag{1.10}$$

$$Int(C) = \{x \in \mathbb{R}^n | h(x) > 0\}. \tag{1.11}$$

Also assume that we have a nonlinear control affine system with the dynamics

$$\dot{x} = f(x) + g(x)u(x). \tag{1.12}$$

Then, we say $h(x)$ is a *control barrier function* if there exist an extended class $\mathcal{K}_\infty$ function α such that

$$L_f h(x) + L_g h(x)u \geq -\alpha(h(x)) \tag{1.13}$$

for all $x \in \mathcal{D}$. Note $L_f h(x)$ and $L_g h(x)$ are Lie-derivative of $h(x)$, e.g., $L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$ and $L_g h(x) = \frac{\partial h(x)}{\partial x} g(x)$. Importantly, the existence of such *control barrier function* $h(x)$ implies the forward invariance of set $\mathcal{C}$ and hence the safety of the system (4). Moreover, the set $\mathcal{C}$ is also asymptotically stable in the domain $\mathcal{D}$. This means even if the system accidentally leaves the set $\mathcal{C}$ and enters $\mathcal{D}\backslash\mathcal{C}$, there exist a controller that brings the system back to $\mathcal{C}$.

Now, define the set of all control that can render $\mathcal{C}$ forward invariant as

$$U_{cbf}(x) = \{u \in U \in \mathbb{R}^m : L_f h(x) + L_g h(x)u \geq -\alpha(h(x))\}. \tag{1.14}$$

As seen above, control barrier functions give the necessary and sufficient conditions for safety. With this established, we would like to know how one can apply these results when designing controllers for safety critical systems. The state-of-the-art approach is to use an

optimization based controller with constraints that reflects the safety conditions provided by the CBF. In particular, the framework assumes we have access to a nominal controller $\tilde{u}$ that does not guarantee safety and attempts to find a safe control input $u \in U_{cbf}$ that is most similar to the $\tilde{u}$:

$$u(x) = \underset{u \in \mathbb{R}^m}{\operatorname{argmax}} \ \frac{1}{2}\|\tilde{u}(x) - u\|^2$$
$$s.t.\ L_f h(x) + L_g h(x) u \geq -\alpha(h(x))$$

Note this optimization yields a minimally invasive controller with respect to the nominal controller since it simply projects the nominal input to the safe input space U_{cbf}. This has the advantage of not deviating too much from the original controller and retains a certain extent of performance guarantee. This framework is referred to as the CBF-QP since it involves a quadratic programming problem and has constraints derived from the control barrier function.

1.2.3 Conclusion

In summary, we wish to leverage ideas from the above areas to help analyze the safety aspect of autonomous systems. Moreover, we propose several methods to enhance system safety that are certifiable using tools and theories derived in these fields. We hope that methods developed in this thesis will provide an incremental improvement along the direction of developing systems that are both safe and efficient. We see this as an important mean of creating a world filled with highly autonomous robots.

1.3 Contributions

The main contributions of this work are:

- An iterative algorithm to solve a multi-player reach-avoid game problem with safety guarantee from the perspectives of attackers.
- A method to perform target re-association after long-term occlusion via a combination of tools from reachability analysis and Bayesian inference.
- A general framework and analysis on learning control barrier functions from user-specify safety requirements for systems with control input constraints.

2. MULTIPLAYER REACH-AVOID GAME IN DYNAMIC ENVIRONMENT

2.1 Introduction

With the increasing acceptance of using fully autonomous system in areas traditionally handled by human, applications such as autonomous driving and drone deliveries are drawing ever more attention. Safe motion planning is obviously an important problem that must be addressed in order for these systems to fully function.

Reachability analysis is a powerful tool to analyze a collection of system trajectories at the same time, and enable one to identify states that adhere to constraints despite disturbances, hence safety guarantees are provided by employing reachability analysis. In [1], a differential game between two Dubins vehicles was posed using a closed-loop formulation and a backward reachable set was obtained by solving a Hamilton-Jacobi-Isaacs PDE. The idea of safe motion planning also plays a significant role in more complex applications such as reach-avoid games and pursuit-evasion games [29],[30],[31]. A reach-avoid game is a game in which an attacker wishes to reach the target region while avoiding a defender with opposite objective, namely to capture or delay the attacker from winning [32] (see section 2.2 for precise definition). The possibilities of having time-varying dynamics, targets, and constraints in a reach-avoid differential game are addressed by modifying the HJI PDE into a double-variational inequality [32] or using a single-variational inequality but requires the state-space to be augmented with an additional time dimension [33]. In [34], [35] the approach was extended to solve a Capture-the-flag game that involves multiple complex, competing objectives. The game was solved as a zero-sum differential game and reachability analysis was used to compute winning regions for each player. The solution to this kind of formulation is minimum time-to-reach functions, which have been proven to be equivalent to the viscosity solution to a HJB equation [36], and can be obtained using level set methods, as shown in [6].

However, the speed of solving such HJ equations is constrained by the infamous "curse of dimensionality"; as the number of dimensions increases, the computation complexity scales in an exponential fashion. This leads to the idea of posing the reach-avoid game as an open-

loop formulation, which allows one to retreat to solving only low dimensional HJB equations for players individually instead of a HJI equation for the joint system [37]. However, the solution (path) found using this method is very conservative to towards attackers and the method can become impractical when the number of defenders increases or for certain initial configurations where defenders are placed between attackers and the goal.

Reach-avoid games with multiple players have also been studied. However, additional vehicles means additional dimensions, so most work involving multi-agent do not solve the problem in the joint configuration space containing states of all agents [38],[39],[40]. Instead, different heuristics or schemes are used to circumvent the computation limitation; agents' motions are planned sequentially based on a predetermined priority in [41], and a maximum matching approach is taken to determine pairwise outcomes between agents in [42]. In this paper, we extend the work in [37] and set out to find a control sequence that allows each attacking agent to reach the target amidst environment changes, motion from other attackers, and disturbance from defenders.

2.1.1 Information Pattern

Since the actions of the agent can affect those of the opposing agents, the value of the game is affected by the order of control and the information pattern chosen. Therefore, it is important to define the information pattern of the game in order to avoid indefinite second-guessing between agents [43]. There are four basic types of information patterns, namely open-loop, state-feedback, non-anticipative strategies, and anticipative strategies [1]. Open-loop strategy is when both agents choose their entire sequences of control inputs u(t) and d(t) for all t in $[0,t_f]$, without knowledge of each other's control. State-feedback strategy refers to when both agents pick their control inputs based on knowledge of the current state of the joint system trajectory. Non-anticipative strategy means agent B not only has access to the current state of both agents, but also the current input of the opposing agent. This gives a slight advantage to agent B and the solution, if found, is conservative towards agent A. Note that agent B must declare its strategy in advance before agent A selects its current input, so agent A is able to determine B's response to any input choice of its own. In anticipative

strategy, agent B has full knowledge of A's control sequence for $t \in [0,\ t_f]$. This gives the most advantage to agent B and essentially causes agent A to run open-loop.

2.2 Problem Formulation

In this section we extend the two-player open-loop game in [37] to a game between two teams with competing objectives. Consider a reach-avoid game with two teams of players with opposite objectives, namely the attackers and the defenders. Let $\mathcal{A} = \{a_1, ..., a_n\}$ denote a team of n attackers and $\mathcal{B} = \{b_1, ..., b_m\}$ denote a team of m defenders. The attackers' winning objective is to reach the goal without being captured by the defenders, and the defenders' objective is to collide into the attackers or delay them from reaching their goal indefinitely.

We also do not assume that the obstacles move in a predictable manner and allow emergence of previously unseen obstacles. In particular, we are interested in finding the minimum-time path for each attacker to reach their goal while avoiding capture by any defender as well as collision between attackers.

The game is being played out in an open region $\Omega \subset \mathbb{R}^2$, which represents a set of points agents are allowed to occupy. Obstacles are denoted as Ω^C and are points that agents may not occupy. The goal region for attacker a_{i} is denoted by closed target set $\tau_{\mathrm{i}} \subset \Omega$. The attackers are modeled using the Dubin's car model, and the state of a single attacker a_{i} is represented by $X_{a\mathrm{i}} = [x_{a\mathrm{i}}, \theta_{a\mathrm{i}}]^T \in \Omega \times [0, 2\boldsymbol{\pi})$, where $x_{a\mathrm{i}} = [p_{x_{a\mathrm{i}}}, p_{y_{a\mathrm{i}}}]$ and $\theta_{a\mathrm{i}}$ are the planar positions and heading of attacker a_{i}, respectively. We also assume defenders are point vehicles free to move in any directions to reflect the fact that we have little information about their dynamics, e.g. defender b_{j}'s state is $X_{b\mathrm{j}} = [x_{b\mathrm{j}}]^T = [p_{x_{b\mathrm{j}}}, p_{y_{b\mathrm{j}}}]^T \in \Omega$. The dynamics are then

$$\dot{X}_{a\mathrm{i}} = f_{a\mathrm{i}}(X_{a\mathrm{i}}(t), u_{\mathrm{i}}(t)) = \begin{bmatrix} v_a \cos\theta_{a\mathrm{i}} \\ v_a \sin\theta_{a\mathrm{i}} \\ u_{\mathrm{i}} \end{bmatrix},\ u_{\mathrm{i}} \in \mathbb{U} \tag{2.1}$$

$$\dot{X}_{b\mathrm{j}}(t) = f_{b\mathrm{j}}(X_{b\mathrm{j}}(t), d_{\mathrm{j}}(t)) = v_b d_{\mathrm{j}}(t),\ d_{\mathrm{j}} \in \mathbb{D} \tag{2.2}$$

where $v_a, v_b, u_{\mathrm{i}}, d_{\mathrm{j}}$ are the speed and individual control inputs of attackers and defenders respectively. We let the sets of admissible control $\mathbb{U} = [u_{min}\ u_{max}]$ and $\mathbb{D}$ be a set of all unit vectors in $\mathbb{R}^2$. Both $f_{ai}, f_{b\mathrm{j}}$ are assumed to be uniformly continuous, bounded, and Lipschitz continuous.

2.2.1 Multi-player Open-Loop Game Formulation

By extending the open-loop formulation proposed in [37] we describe the cost function of the game between each attacker a_{i} and all defenders as time-to-reach function

$$J_{\mathrm{i}}(x^0, u_{\mathrm{i}}, \boldsymbol{d}) = \inf\{t | x_{a\mathrm{i}}(t) = \tau_{\mathrm{i}}, x_{a\mathrm{i}}(s) \neq x_{b\mathrm{j}}(s), \\ \mathrm{j} = 1, ..., m, \forall s \leq t\} \tag{2.3}$$

where $\boldsymbol{d}$ denotes a collection of all defenders' inputs $\{d_1, ..., d_m\}$.

In an open-loop game, the attacker picks its entire control sequence at $t = 0$, with the assumption that the defender will defend with its optimal strategy for all $t \geq 0$. This is the most conservative information pattern towards attackers and it yields an upper bound of the value function, namely the upper value of the game [37]. This upper value for a game between attacker a_{i} and a team of defenders $\mathcal{B} = \{b_1, ..., b_m\}$ is defined as

$$v_{\mathrm{i}}(x^0) = \inf_{u \in \mathbb{U}} \sup_{d \in \mathbb{D}} J_{\mathrm{i}}(x^0, u_{\mathrm{i}}, \boldsymbol{d}) \tag{2.4}$$

A safe path (and its corresponding control sequence) is said to be found if such upper value is finite, e.g. agent a_{i} can reach τ_{i} in finite time. Even if the upper value $v_{\mathrm{i}}(x^0) = \infty$, it only guarantees an existence of controls for defenders to cause a collision if all defenders in the team act optimally. In reality, some defenders might not actually take the optimal actions for various reasons, which means attacker a_{i} still has a chance to reach its goal. Furthermore, collision-free control sequences for attackers might still be found using a less conservative information pattern, such as non-anticipative feedback strategy.

Similar to [37], we say a point $y \in \Omega$ is i-safe-reachable if it can be reached by a_{i} within finite time t and no capture by any defender can happen before time t, or formally

$\{y|\,\exists u_{\mathrm{i}}, t \geq 0, x_{a\mathrm{i}}(t) = y, x_{a\mathrm{i}}(s) \neq x_{b\mathrm{j}}(s), \mathrm{j} = 1, ..., m, \forall s \leq t\}$. The trajectory $x_{a\mathrm{i}}(\cdot)$ is referred to as i-safe-reachable path. With this, the collection of i-safe-reachable points for a_{i} is known as the i-safe-reachable set,

$$S_{\mathrm{i}} = \{y \in \Omega \mid y \text{ is a i-safe-reachable point}\}$$

The minimum time-to-reach function for an attacker a_{i}, $\varphi_{\mathrm{i}} : \Omega \to \mathbb{R}$ can now be defined as

$$\varphi_{\mathrm{i}}(y) = \min\{t \,|x_{a\mathrm{i}}(t) = y, x_{a\mathrm{i}}(s) \in S_{\mathrm{i}}, \forall s \leq t\}. \tag{2.5}$$

The minimum time-to-reach function for a defender b_{j}, $\psi_{\mathrm{j}} : \Omega \to \mathbb{R}$ is defined as

$$\psi_{\mathrm{j}}(y) = \min\{t \,|x_{b\mathrm{j}}(t) = y, x_{b\mathrm{j}}(s) \in \Omega, \forall s \leq t\}. \tag{2.6}$$

According to [37] (Theorem 1), the upper value function for a_{i} is proven to be:

$$v_{\mathrm{i}}(x^0) = \min\{\varphi_{\mathrm{i}}(y) \mid y \in \tau_{\mathrm{i}}\}$$

Note that so far game domain is represented as a static environment Ω and no changes in the environment have been considered. Since these changes, such as the movement of obstacles, can be random, we cannot model them during the planning stage and must act on a sense-and-react basis. In addition, we assume each agent is able to obtain an updated map of the environment at a fixed frequency from sensor updates. To accommodate the assumptions above, we denote the dynamic domain as $\Omega(kT)$, where T is the sensing interval. We use Ω_{kT} to denote $\Omega(kT)$ in the rest of this paper for simplicity. The **problem of interest** is to find the minimum-time optimal path for each attacker a_{i} to reach its target τ_{i} while guaranteeing $x_{a\mathrm{i}}(s) \neq x_{b\mathrm{j}}(s), \forall \mathrm{j} = 1, ..., m, \forall s \leq t_f$ (hence collision avoidance), under a dynamic environment Ω_{kT}.

The minimum time-to-reach function is known to be equivalent to the viscosity solution to a particular Hamilton-Jacobi-Bellman (HJB) equation [44]. We use $\varphi_{\mathrm{i},k}(y)$ to denote the

minimum time-to-reach function for a_i at time-step k. Note that the definition of $\varphi_{\mathrm{i},k}(y)$ in (2.5) satisfies the below HJB equation

$$-\min_{u_\mathrm{i}\in U}\{\nabla\varphi_{\mathrm{i},k}(y)\cdot f_a((y,\theta),u)\}=1,\forall y\in \boldsymbol{S}_{\mathrm{i},k}\backslash\{x_{a\mathrm{i}}^0\} \tag{2.7}$$

where the boundary conditions are

$$\varphi_{\mathrm{i},k}(x_{a_\mathrm{i}}^{\,0})=0, \tag{2.8}$$

$$\varphi_{\mathrm{i},k}(y)=\infty, y\in\Omega_{kT}\backslash\boldsymbol{S}_{\mathrm{i},k} \tag{2.9}$$

where $\boldsymbol{S}_{\mathrm{i},k}$ is the intersection of safe sets of a_i relative to all b_j's, $\boldsymbol{S}_\mathrm{i}=S_{\mathrm{i}1}\cap\ldots\cap S_{\mathrm{i}m}$, at timestep k. The second boundary conditions reflects that the minimum time-to-reach of a point not belong in $\boldsymbol{S}_{\mathrm{i},k}$ is infinity since there exist a defender who can reach such point before a_i. The solution to our problem then essentially translates to solving equations (2.7)-(2.9) at each timestep k. Once $\varphi_{\mathrm{i},k}(y)$ is known, the optimal path at timestep k is obtained by following the gradient of $\varphi_{\mathrm{i},k}(y)$.

2.3 Methodology

If we replace the anisotropic dynamics[1] in (2.7) with isotropic dynamics (e.g. $\tilde{f}_a(y,\hat{u})$), then equation (2.7)-(2.9) can be solved by a slight modification of the modified Fast Marching Method (FMM) in [37]. By solving this equation for each attacker against all m defenders and taking the intersection of m safe sets, we can obtain the solution for the multi-player open-loop game from attackers' perspectives. This process is repeated at every time-step after new measurements of the environment is received. However, we note that simply repeating the algorithm in [37] k times does not completely solve our problem and there are several issues that must be addressed. First, we must account for the practical non-holonomic constraints (e.g. minimum turn-radius) for the attackers since FMM assumes an

[1]↑anisotropic dynamics is when the vehicle motion depends not only on the positions, but also the direction/heading.

isotropic dynamic. Second, when multiple agents are involved, collision avoidance needs to be enforced not only between attackers and defenders, but also among attackers. To address these issues, we incorporate a virtual obstacle method and the idea of forward reachable set into the iterative framework, as discussed in this section.

2.3.1 Modified Fast Marching Method for Multi-Agent Game

FMM was originally proposed to address the problem of wavefront propagation and solve the Eikonal equation [45]:

$$1 = F(x)|\nabla T(x)| \tag{2.10}$$

where $F(x)$ is the propagation speed and $T(x)$ is the time-to-reach function. Note that (2.10) has the same form as (2.7) and if we let $F(x)$ equal the isotropic dynamics $\tilde{f}_a(y, \hat{u})$), then $T(x)$ is exactly the minimum time-to-reach function $\varphi(x)$ for attackers in (2.7). Hence, we use $\varphi(x)$ in place of $T(x)$ when describing the FMM scheme below.

To solve this continuous problem numerically, one must do so on a discretized domain (grid). We first give a few definition of the labels used to categorize the points on the grid. *Alive*-points whose φ values have been computed. *Trial*-the point currently being processed. *Near*-points that are in neighborhood of the Trial point. *Far*-all points that are not within Trial's neighborhood. We now describe the procedure of the corresponding numerical scheme. To initialize, a starting point (u, v) on the grid is initiated as $\varphi(u, v) = 0$, indicating the time-to-reach of this point is zero. We then tag this point as Alive, and the points one grid point away as Near. All other points are tagged as far. Then we compute the value of all Near points using the following finite difference approximation of the Eikonal equation

$$\begin{aligned} &\frac{F(x_{u,v})}{h}[(\varphi_{u,v} - min\{\varphi_{u\pm1,v}, \varphi_{u,v}\})^2 + \\ &(\varphi_{u,v} - min\{\varphi_{u,v\pm1}, \varphi_{u,v}\})^2]^{1/2} = 1 \end{aligned} \tag{2.11}$$

where h is the grid resolution. The solution $\varphi^*(u, v)$ can then be obtained by solving equation (2.11).

The rest of the algorithm follows the following loop:

1. Set the values of all Far point as ∞
2. Set Trial to be the point in Near with the smallest value
3. Add Trial point to Alive list and remove it from Near
4. Set all the neighbors of the Trial point to be Near (if the points are not Alive). If any of such neighboring points is in Far we remove it from the Far list
5. Compute the values of all new Near points via solving the quadratic equation (2.11).
6. Repeat steps 2-5 until all points are Alive.

Note that to obtain the safe set S_{i}, the authors in [37] modified the algorithm by adding an intermediate step after step 3:

3.5) if at Trial point $\varphi_{\mathrm{i}}(u,v) > \min_{\mathrm{j}}\{\psi_{\mathrm{j}}(u,v)\}$, set $\varphi_{\mathrm{i}}(u,v) = \infty$.

This modification is based on the fact that if the time-to-reach for attacker a_{i} is greater than the minimum of time-to-reach of all defenders $b_{\mathrm{j}=1,\ldots,\mathrm{j}=m}$, then such point should not belong to the safe set S_{i}. Fig. 2.1 shows a simple illustration of resulting safe path and safe set computed using such method in MATLAB [46].

2.3.2 The Virtual Obstacles Approach

Now we further modified the algorithm to incorporate for turning radius constraints imposed on attackers. Similar to [47], we place virtual obstacles around the vehicle to create an effect of having a minimum turning radius. This idea is similar to the way configuration space deals with dynamic constraints, but we only need to do so locally. For example, in the case of a Dubin's car with the constraint of max. turning rate u_{max} on a 2-D plane, we place two circular obstacles with radii of $\frac{1}{u_{max}}$ beside the vehicle, as well as one behind it to prevent any reverse motion, as shown in Fig. 2.2. Such virtual obstacles cause the FMM method to generate a path whose segment over next time-step respects the turning radius, hence satisfying the constraints locally. As this process is repeated at every time-step due

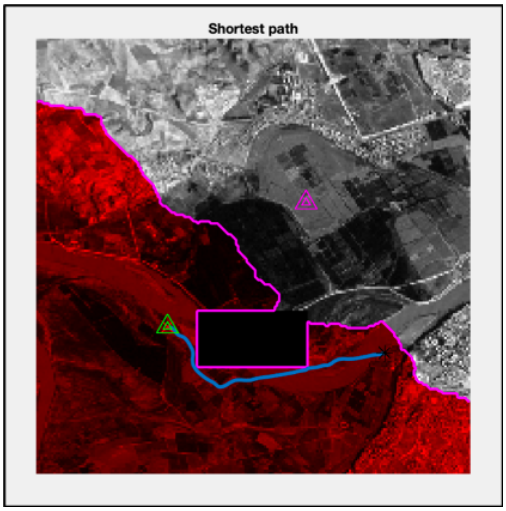


Figure 2.1. Solution of open-loop game via FMM[46]: Magenta triangle = defender initial state, green triangle = attacker initial state, black rectangle = obstacle, red area = attacker's safe set S.

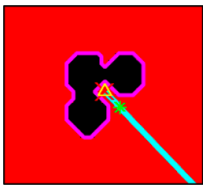


Figure 2.2. Virtual obstacle method for Dubins car model with minimum turn radius on 2D plane.

to the nature of our iterative scheme (elaborated below), the entire path executed by the vehicle will satisfy the constraints. We note that this approach is particularly suitable to our real-time algorithm since it introduces little computation burden.

2.3.3 Forward Reachable Set for Inter-Attacker Collisions

In multi-agent scenarios, it is necessary to also address inter-attacker collision. To do this we exploit the idea of forward reachable set (FRS). In reachability analysis, the FRS of a system is the set of states that can be reached from the initial state after time t_f. Computing the FRS can be, however, a computationally expensive process, and using the full FRS in avoidance application is often too conservative for the planner since it considers every possible states reachable within the planning horizon. Nevertheless, this is useful to

our application since we are only interested in the FRS of one time step, e.g. $t_f = T$ (T is sensing interval). By planning around the T-FRS of the other attacker we ensure no inter-attacker collision can happen. We also note that the FRS can be computed offline given the attackers' dynamics.

2.3.4 Replanning via Iterative Open-Loop Formulation

In this section, we discuss the considerations for extending the open-loop reach-avoid game in [37] to an iterative open-loop formulation. This new formulation is in between the open-loop strategy and continuous state-feedback strategy, which yields a good balance between optimality and computation complexity. Furthermore, changes in both the environment and objectives of the defender can be addressed through its iterative nature.

Instead of computing the safe time-optimal path only once at the beginning of the game, we compute this path at every sampling instance k using the current state (e.g. most recent positions of the defender, obstacles, and other attackers) as the initial condition. Upon obtaining the solution, we only execute the control sequence for a sample period T and discard rest. This process is repeated until agent a_{i} reaches τ_{i}, in an MPC-like fashion. This iterative approach implicitly addresses the issues of both unpredictable moving obstacles and defenders. Further, this allows us to find paths that do not exist initially and only open up as the environment changes. Due to the re-planning nature of our method, the path ultimately executed can significantly deviate from the initial planned path. Note that if an attacker begins in a losing configuration, a safe path will not be found until either the defender deviates from its optimal trajectory or the environment changes. The iterative scheme is summarized in Algorithm 1.

2.4 Simulation Results

This section presents several simulations demonstrating the validity of our iterative scheme in both single and multi-agent settings.

Algorithm 1: Multi-agent Collision Avoidance via Iterative Open Loop Formulation.

Result: Iterative method for Reach-Avoid Game

`/* Start algorithm */`

for $k = 1, ..., N_s$ *(sampling time-steps)* **do**

  `/* at every fixed time-interval */`

  Record the current locations of obstacles and defenders

  for i $= 1, ..., N_a$*(number of attacker)* **do**

    `/* check where other attackers are */`

    Record current states of attackers j $= 1...N_a$, j $\neq$ i

    Augment other attackers with precomputed FRS

    Apply virtual obstacle method to attacker a_{i}

    Compute φ_{i} for attacker a_{i} using modified FMM and determine $S_{\mathrm{i},k}$

    if *the goal is not in the safe set* $S_{\mathrm{i},k}$ *of* a_{i} **then**

      `/* attacker cannot win unless defenders act sub-optimally */`

      Set temporary goal for a_{i} to be the point in $S_{\mathrm{i},k}$ with the lowest time-to-reach value from the actual goal

    end

    Extract optimal path sequence from calculated φ_{i}

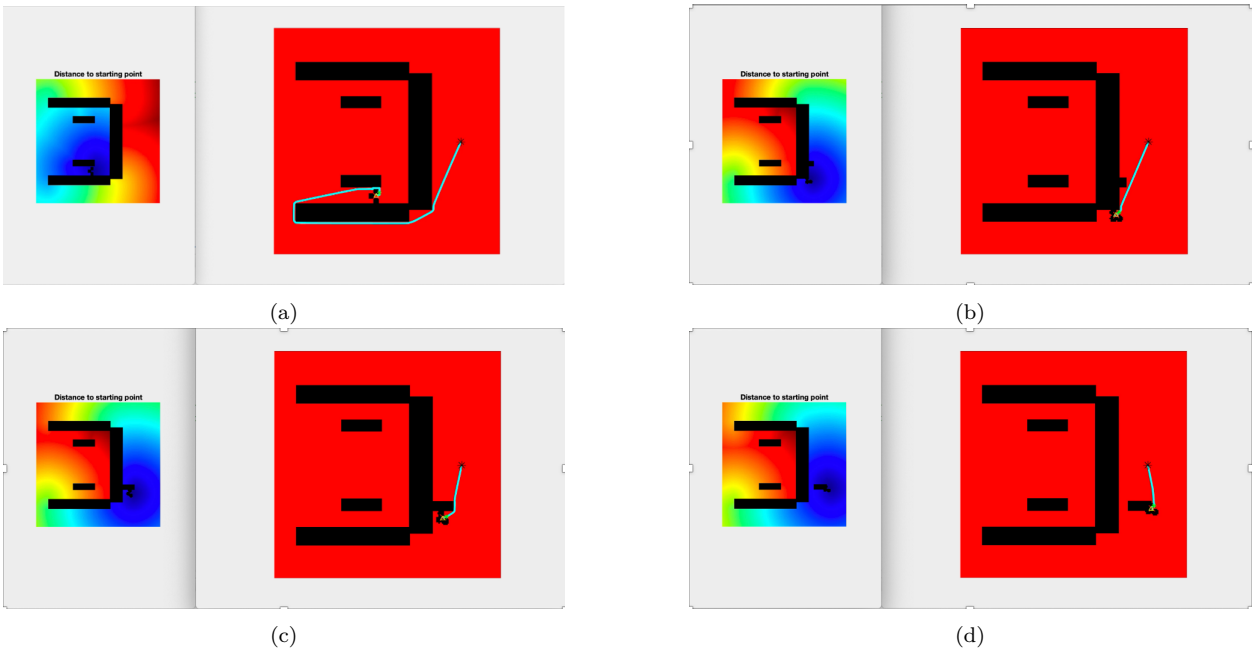
    Update state of a_{i} for 1 sampling period T

  end

end

2.4.1 Replanning Around Unexpected Obstacles

First we demonstrate the replanning ability of the proposed algorithm in an environment with unpredictable obstacles. Fig. 2.3 shows the vehicle navigating to goal while encountering a unplanned obstacle. The vehicle is able to update the path as soon as the obstacle appears and successfully reach its goal.



(a) (b) (c) (d)

Figure 2.3. Vehicle avoids unexpected obstacles via re-planning

.

2.4.2 Differences Between Initial and Executed Path

Fig. 2.4 shows a comparison between the planned path and the actual path executed by the vehicle. As we already know, the vehicle must re-plan to avoid the unexpected obstacle, so the two paths differ significantly in such area. In addition, the vehicle discovers a shorter path by making a U-turn near the start; such path was infeasible initially due to constraints but opens up as the vehicle moves upward. We also observe that the executed path is a lot smoother compared to the initial path. This is because the virtual obstacle method ensures the trajectory of the vehicle satisfies the dynamic constraints. We note that the initial path

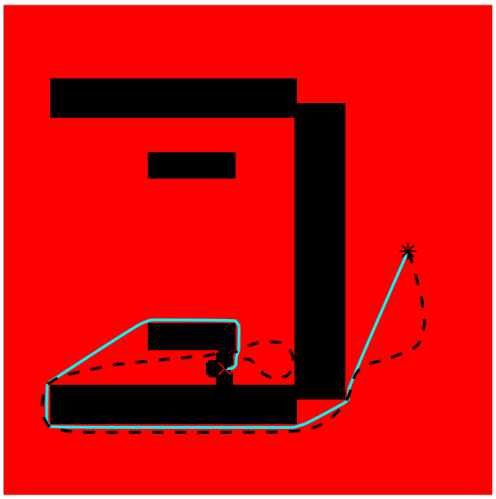


Figure 2.4. Comparison between actual and initial paths.

contains unrealistic sharp turns and attempts to make the vehicle stay as close to the walls as possible since the constraints are not imposed globally.

2.4.3 Multi-agent Scenario

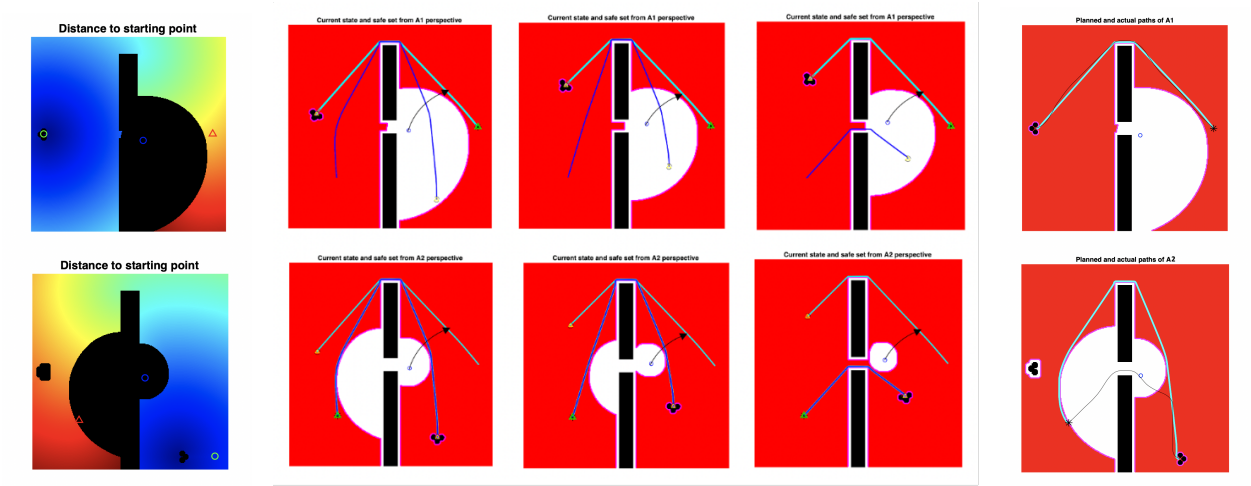
We also extend our algorithm to a multi-agent setting that involve several attackers. We simulate two scenarios, one with two attackers and one defender, the other with three attackers and two defenders. The first scenario is depicted in Fig. 2.5, where attackers a_1 and a_2 begin on the left and right side, respectively. The objective of the attackers is to reach their goals on the other side while avoiding the defender in the middle. Fig. 2.5a shows the values of the minimum time-to-reach function $\varphi_{\mathrm{i}}(y)$ across the domain, with blue as the lowest and red as the highest value. The dark regions represent obstacles and points that are reachable by defenders before attackers (hence unsafe region $\Omega \setminus S$). The safe path is computed via a simple gradient descent method.

Note that in Fig. 2.5a (bottom figure), the goal location is contained within the unsafe region, meaning that if the defender takes optimal action against attacker 2 (in this case the optimal action is to race to the goal and wait for attacker 2), then there does not exist a safe path for attacker 2 to reach the goal. The goal is contained within the safe set from attacker 1's perspective, so a_1 has a way to reach the goal regardless of what the defender does. In

Fig. 2.5b, we let the defender pursue attacker a_1 by trying to intercept it at a point along its path (cyan). Although this is near-optimal in terms of pursuing a_1, it is sub-optimal in terms of defending against a_2. This leads to the goal location eventually 'leaving' the unsafe set for a_2 and become contained within attacker 2's safe set, creating a path that leads to the goal safely. Indeed, we can see that a_1 is forced to follow the initial path as the defender blocks the gap for a long enough duration before trying to intercept it (first row). However, since the goal was contained in a_1's safe set, the defender cannot win against a_1 no matter how hard it tries. The black arrows in Fig. 2.5b indicate the actions of the defender. As time evolves the defender moves away from the gap, causing a new path to be found (Fig. 2.5b third column) for attacker a_2. Because of such action from the defender, attacker 2 is able to take a more direct path compared to what's planned initially; the unsafe set for a_2 is receding away from the direct path to goal. Fig. 2.5c compares the initial paths (cyan) and actual paths (black) of two attackers. We also want to point out that under certain situations the open-loop solution is equivalent to the solution given by state-feedback formulation and the control of the defender is irrelevant [48].

Fig. 2.6a-2.6c show a second simulation illustrating that our formulation can handle situations with multiple attackers and multiple defenders. In particular, we want to highlight two points in this example. First, we demonstrate how inter-attacker collision is avoided. Secondly, we show how the outcome of the game can change as a result of sub-optimal actions of players. Here the attackers start at the end of the "tunnels" and want to reach the goals on the opposite side of their initial positions on the domain. Attacker a_1 (left side) has its goal located on the top right area, a_2's goal is on the bottom left, and a_3's goal is directly below it. Two defenders' initial positions are set such that the attackers are force to go through the gap between tunnels (d_1 at bottom-left and d_2 at top-right). Initially, both attacker a_1 and a_2's goal are not in their respective safe set, so they can only navigate to temporary goals at best. Note that this means a_1 and a_2 are expected to lose the game against d_2 and d_1, respectively, given optimal play. Under this situation, a game of degree is being played between attackers and defenders and the value function is the terminal distance between the attacker and its corresponding goal [49]. Therefore, the aforementioned temporary goal for each attacker is set to minimize such distance. Again, the defenders here attempt to pursue

the attackers instead of racing to the goal, which are sub-optimal but reasonable actions to take in reality as they may not have full knowledge of the goal locations. These actions opens up paths for the initially doomed attackers and changes the game's outcome. With the path to each goal eventually found, one challenge remains for the attacking team; the attackers must still avoid other attackers since they are forced to move towards the common area at the center. By employing the FRS method discussed above, we successfully prevent inter-attacker collision, as shown in Fig. 2.6b-2.6c. Fig. 2.7a-2.7c show the comparison between the initial and final paths taken by the attackers. Note that in Fig. 2.7c attacker a_3 is forced to a go-around maneuver in the center to prevent collision with a_2.

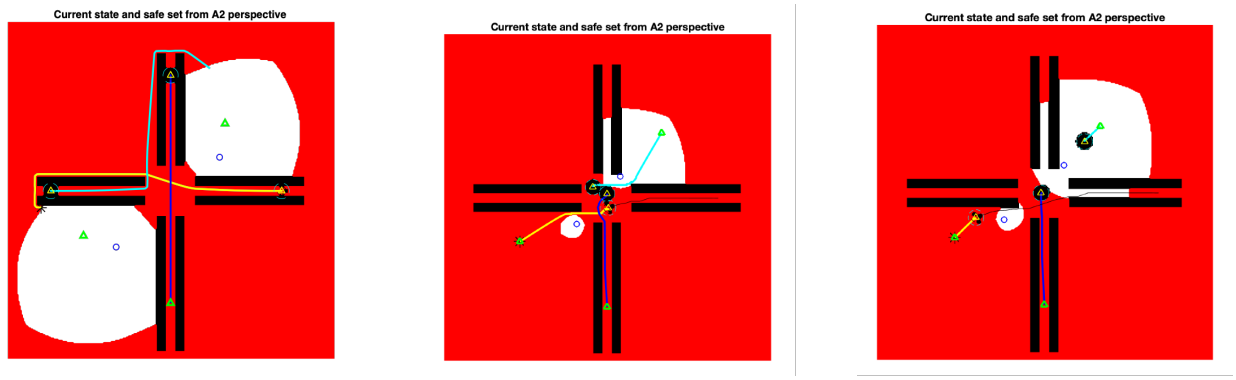


(a) (b) (c)

Figure 2.5. Multi-agent scenario with two attackers and single defender: (a) initial configuration of agents (green circle=attacker start point, blue circle=defender start point, red triangle=attacker goal) and time-to-reach value map. (b) Positions of agents over time (Left to right) and evolution of safe set (red region) from attacker a_1 and a_2's perspectives (top and bottom row respectively). Green triangle = goal, blue circle = defender, yellow triangle = attacker. (c) Comparison of executed path and initial path for both attackers (black=actual, cyan=initial).

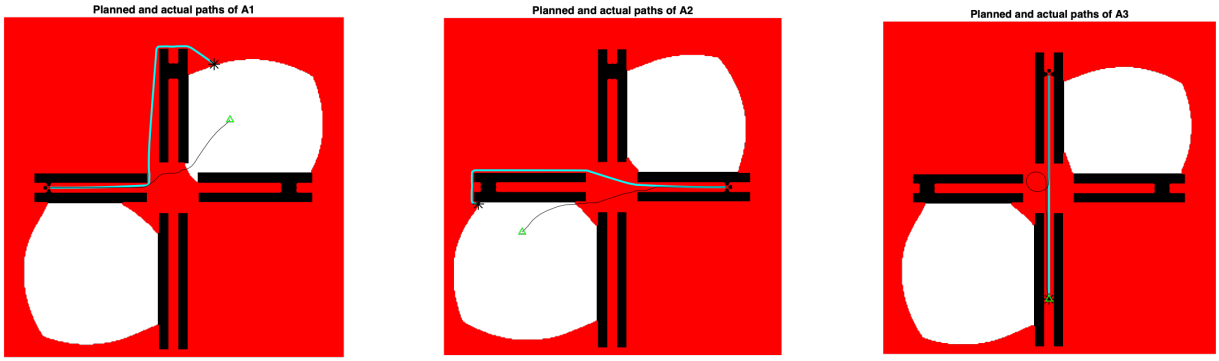
2.4.4 Comparison with Existing Methods

Our iterative open-loop formulation takes a middle-ground between the open-loop formulation (where solution only depends on initial conditions) and the state-feedback/non-



(a) (b) (c)

Figure 2.6. Multi-agent scenario with three attackers and two defenders: (Left to right) positions of agents over time (Green triangle=goal, blue circle=defender, yellow triangles=attacker). Only the safe set from attacker a_2's perspective is shown.



(a) (b) (c)

Figure 2.7. Comparison between initial and executed paths for attackers a_1, a_2, and a_3 (respectively shown in (a), (b), and (c)).

anticipative information pattern (where the decision of defender depends on instantaneous input of attackers). The resulting path obtained is less conservative than that of the open-loop formulation, and is less computation intensive compared to the non-anticipative strategy. Further, the extent of conservativeness can be adjusted through changing the update interval. Our method can also handle different dynamic constraints, as long as the shape of corresponding virtual obstacles can be computed analytically. In terms of scalability, the iterative scheme has advantage over the other two strategies as additional agents can be in-

troduced without much extra computation load. Although the normal open-loop formulation can technically be applied to settings with more agents, the over-conservative nature of the formulation may quickly prevent us from finding feasible paths, even when such paths exist. The non-anticipative strategy requires solving a high-dimensional HJI PDE, so introducing more agents is not practical.

2.5 Conclusion

In this work, an iterative safe motion planning algorithm is presented and is shown to work well in hostile, uncertain environments via simulations. We extend the open-loop formulation of a reach-avoid game in [37] to an MPC-like scheme, in which the FMM generated path is recomputed after every time-step. We also propose a way to allow FMM to work with non-holonomic vehicles and ensures the resulting path satisfies the constraints of a Dubins car. In addition, we address the problem of inter-attacker collision via the use of forward reachable set. We show that the algorithm generate feasible paths that are less conservative than that of the open-loop formulation under multi-agent settings. More importantly, this iterative framework allows the outcome of the game to be changed when optimal play from any side does not take place. In the future, we could assign a probability of each defender being a direct threat to each attacker, with the assumption that each defender can only be a threat to one attacker at any instant. This could further reduce the conservativeness (towards attackers) of the solution and help discover even more efficient paths. Alternative methods to address motion constraints such as the modified gradient control law in [50] could work well with the current framework. Lastly, incorporating cooperation [51],[52] into such reach-avoid game (for both attackers and defenders) is also a very interesting future research direction.

3. TARGET REASSOCIATION AFTER LONG-TERM OCCLUSION VIA REACHABILITY AND BAYESIAN INFERENCE

3.1 Introduction

With the increasing popularity of autonomous robotic systems in areas ranging from autonomous driving in urban areas to military defense, the development of a robust tracking system is more important than ever. The Multi-target Target Tracking (MTT) estimates the location and the number of targets in noisy observations. The traditional method such as nearest neighbor filter (NNF)[53], joint probabilistic data association filter (JPDAF)[54][55][56], and multiple hypothesis tracking (MHT)[57][58] utilize a measurement-to-track association approach, to track multiple targets. The idea is assigning one measurement to one trajectory at each time step to map the multiple target tracking into single target tracking.

Recently, the probability hypothesis density (PHD) filter [59][60] has been proposed as an alternative for MTT. It is a practical solution for Bayesian multi-target filter which propagates the first order moment of the multi-target posterior as an approximation of the full multi-target posterior density. The Gaussian Mixture PHD (GM-PHD) filter [61][62] has been proposed as an implementation of the PHD filter which provides a close form solution. PHD filter estimates the number and the state of the target simultaneously without doing measurement-to-track association. However, data association is an essential subproblem that influences the robustness of a tracking algorithm. Therefore, there exist some approaches to provide identity management method for completing target association between each time step, while some employs a penalization scheme whenever a target is prone to violate the one-to-one assumption [63].

Consider a tracking failure example, where the system has lost track of one or more targets for a prolonged period of time due to occlusion. Many works have been developed regarding the problem of tracking under momentary (partial/total) occlusion, where a target might be blocked by an obstacle in the scene. Han et al. [64] use the Kanade-Lucas-Tomasi (KLT)

features as the representation of objects and propose a trajectory estimation algorithm with a weighting function of tracked features as long as the object is not fully occluded. Shu et al. [65] uses the part-based model. It is highly discriminative and robust against appearance changes and occlusions but failed in full occlusion. However, these methods do not consider possible subsequent missed detection which may occur due to occlusion (e.g. obstacles in the environment). Such limitations caught our attention and motivates us to develop the work below.

To the best of our knowledge, very little work has considered an occlusion period long enough for targets to significantly change their trajectories for inter-target avoidance or changes in objective. In this work, we set out to find a method that, given the disappearance locations and reappearance locations, correctly reassociate the targets to their original identities after they have been redetected. We envision that this proposed method can be integrated with an existing tracking algorithm (such as the GM-PHD filter) to improve its robustness as well as accuracy under prolonged occlusions.

The main contribution of this work is twofold. First, we formulate the problem of target identification under occlusion into a reachability problem. Second, we solve the problem of estimating the probability of identity via our approach inspired by [66] that combines Bayesian inference and forward reachable sets (FRSs). Bayesian inference will be used along with measurements of state trajectory and dynamic model to compute the posterior distribution over a set of feasible control inputs. We wish to obtain a probability associated with each discrete $u_{\mathrm{i}} \in [u_{min} u_{max}]$ of the target at the instant before occlusion takes place. This result should assign the highest probability to the most likely u_{i} based on the previously observed behavior. Reachability analysis is then used to determine all the possible reachable states of the target at a future time. In particular, a FRS is computed for every discretized u_{i} using the level set methods, with the initial set set as the state at which the target disappears. Finally we query the re-appearance state and flag all the FRSs that contain such state and record the corresponding u_{i}'s that gave rise to these FRSs. The probability of identity can then be estimated by summing of all the probabilities that correspond to the flagged u_{i}'s.

Note that our method was initially designed to be used in a post-processing step since it requires knowledge of both disappearance locations and reappearance locations of all targets, as well as the periods of disappearance. However, since most computation can be done in advanced and calculating the probability of identities only requires a look-up on the reachable set, the proposed method can be run near real-time and provide results shortly after all targets re-appeared. Again, we hope this will help enhance the performance of the tracker and reduce the odds of having to create new tracks.

3.2 Problem Formulation

Consider arbitrary number of targets moving around in the surveillance area. These targets are presented by a discrete-time dynamical model, which is chosen to be a Dubins model in our experiment. The constant velocity Dubins model treats each vehicle as a particle with state variables (x, y, θ), where (x, y) represent the center of the mass of the vehicle, and θ is the angle between the velocity vector and x-axis. The control input u represents the rate of change of angular speed. The inputs are bounded by a minimum and a maximum value, represented as $u \in [u_{min}, u_{max}]$. Our dynamic model for the system is then given by

$$x(t+h) = x(t) + hv\cos(\theta(t)) \tag{3.1}$$

$$y(t+h) = y(t) + hv\sin(\theta(t)) \tag{3.2}$$

$$\theta(t+h) = \theta(t) + hu(t). \tag{3.3}$$

We also consider a stochastic disturbance $\boldsymbol{w}$ as an exogenous input drawn from a known distribution $\mathcal{W}$ to simulate the noisy measurements. Our discrete time dynamical model is then

$$\boldsymbol{x}(k+1) := f(\boldsymbol{x}(k), \boldsymbol{u}(k)) + \boldsymbol{w}(k) \tag{3.4}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ denotes the system states and $\boldsymbol{u} \in \mathbb{R}^m$ denotes control inputs. We assume that the state variables $\boldsymbol{x}(t)$ are given at each time step in the form of noisy measurements. We

also assumed that $\boldsymbol{w}(t)$ is bounded by an interval $[w_{min}, w_{max}]$. By truncating the distribution $\mathcal{W}$, we get the a θ confidence interval for a fixed θ, such as 0.99.

We note that the inputs $\boldsymbol{u}(t)$ at each time step cannot be directly measured and therefore must be inferred. Another assumption regarding the control is that we assume the control input gradually changes within a reasonable range $[u_{min}, u_{max}]$ between subsequent time steps. This can be represented by the following equation:

$$\boldsymbol{u}(t+h) = \boldsymbol{u}(t) + \epsilon[\boldsymbol{u}_{min}, \boldsymbol{u}_{max}]. \tag{3.5}$$

This indicates that $\boldsymbol{u}(t)$ evolves over time with reasonable turning rate and inline with the practical situation for a vehicle. Here, the parameter $\epsilon \in [0, 1]$. This is also meant to capture the same effect as transition ϵ, which is used in updating the posterior at time t into the prior at time $t + h$.

In our example scenario in Fig. 3.1, three targets are moving around with respect to the constant velocity Dubins model. When a target passes behind the obstacle for a prolonged period of time we say an occlusion is taking place. This means no measurement are being received at all during this period. In a general tracking algorithm and literature this is known as lost track. Even though the target reappears after several seconds, it will just be considered as a new target and a new track is formed. What we want to do is to assign the probabilistic values for the identities of the occluded targets between disappearing and reappearing point.

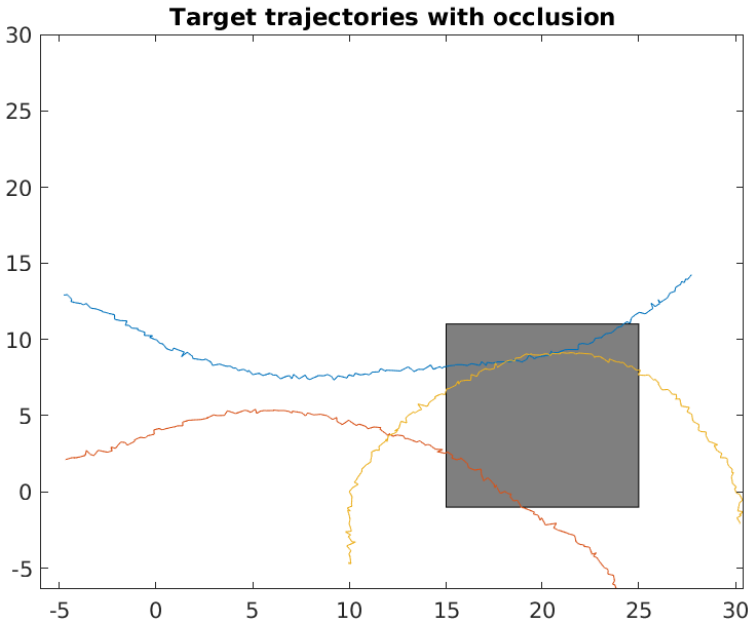


Figure 3.1. Problem Scenario

Below is a summary of the major assumptions we make in this work:

- Vehicle model with state variables $\boldsymbol{x}$, inputs $\boldsymbol{u}$ and exogenous inputs $\boldsymbol{w}$. We assume that the state variables $\boldsymbol{x}$ are known at each time step through noisy measurements with known error distributions, but the inputs $\boldsymbol{u}(t)$ are not measured. At the same time, we assume the exogenous disturbance is drawn from a known distribution ($\mathcal{W}$).
- State measurements are received at some fixed time step. $h > 0$.
- The control input $\boldsymbol{u}$ evolves randomly within a bounded set.

Given this information, at some time instant t_0, when occlusion starts, we wish to know the distribution of possible states $\boldsymbol{x}(t_0 + Nh)$ for some time horizon $N > 0$ (Occlusion period). More specifically, our **problem of interest** is to estimate the probability that an object emerging from occlusion corresponds to the original object of interest, i.e,

$$P(\boldsymbol{x}(t_0 + Nh) = \boldsymbol{x}_{out,\mathrm{i}}). \tag{3.6}$$

Additionally, we wish to compute a "probability identity matrix" that summarizes the identity relationship among targets.

3.3 Methodology

At a high level, our approach is as follows: we first discretize the set of admissible control input and estimate the probability distribution over these inputs at $\boldsymbol{x}(t_0)$; this can be done via standard Bayesian inference. We then compute the reachable sets by propagating states forward using each discretized input as a starting input; this is done via the level set approach [1] and the Level Set toolbox [6] with some minor modifications. Finally we identify the FRSs that contain the out-state and compute the identity probabilities by summing up the control input probabilities with which these reachable sets are associated.

3.3.1 Control Input Estimation Via Bayesian Inference

In this section we talk about how Bayesian inference is used to computed the distribution over $\boldsymbol{U}$ at t_0. At a high level the process is as follows: First assume a prior distribution over $\boldsymbol{U}$ before any measurements are taken. Then compute the posterior distribution using incoming measurement $\boldsymbol{x}(t)$ and update the prior $P(\boldsymbol{u}(t-h))$ into posterior $P(\boldsymbol{u}(t-h)|\boldsymbol{x}(t))$ via Bayes rule. Finally we update the posterior $P(\boldsymbol{u}(t-h)|\boldsymbol{x}(t))$ into a new prior $P(\boldsymbol{u}(t))$ for next iteration. This process is repeated until the prior at t_0 is obtained.

Calculating posterior $P(\boldsymbol{u}(t-h)|\boldsymbol{x}(t))$

We approximate the set of support $\boldsymbol{U}$ for the posterior by selecting finitely many grid points $\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{K-1}$ for some number K. Then associated posterior probabilities for input $\boldsymbol{u}$'s can be represented as

$$(\boldsymbol{u}_0, p_0), (\boldsymbol{u}_1, p_1), \ldots, (\boldsymbol{u}_{K-1}, p_{K-1}) \tag{3.7}$$

where, $p_0, p_1, \ldots$ are the associated probabilities and $\sum_{\mathrm{i}=0}^{K-1} p_{\mathrm{i}} = 1$. We have the un-normalized posterior likelihood

$$\Lambda\left(\boldsymbol{u}(k-1) = \boldsymbol{u}_{\mathrm{j}} | \boldsymbol{x}(t)\right) = P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h))P(\boldsymbol{u}(t-h)). \tag{3.8}$$

Hence, we have, the posterior probability

$$P(\boldsymbol{u}(t-h) = \boldsymbol{u}_{\mathrm{j}}|\boldsymbol{x}(t)) = \frac{\Lambda(\boldsymbol{u}(t-h)) = \boldsymbol{u}_{\mathrm{j}}|\boldsymbol{x}(t))}{\sum_{\mathrm{j}} \Lambda(\boldsymbol{u}(t-h)) = \boldsymbol{u}_{\mathrm{j}}|\boldsymbol{x}(t))} \tag{3.9}$$

where $P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h))$ is computed directly from the model, the measurements $\boldsymbol{x}(t-h)$ and $\boldsymbol{x}(t)$ (and the randomness comes from $\boldsymbol{w} \sim \mathcal{W}$). The next section describes the process of computing the likelihood term in 3.8.

Computing the likelihood $P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h))$

The likelihood $P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h))$ can be written as

$$P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h)) = P\left(f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h)) + \boldsymbol{w}(t-h)|\boldsymbol{u}(t-h)\right) \tag{3.10}$$

where $\boldsymbol{x}(t-h)$ is also known. Probability of arriving at a particular $\boldsymbol{x}(t)$ from given $\boldsymbol{x}(t-h)$ via a $\boldsymbol{u}(t-h)$ can be perceived as the probability of the noise $\boldsymbol{w}$ being equal to $\boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h))$. Then the likelihood can be written as

$$P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h)) = P(\boldsymbol{w} = \boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h))|\boldsymbol{u}(t-h), \boldsymbol{x}(t-h), \boldsymbol{x}(t)). \tag{3.11}$$

Let $\boldsymbol{w}(t)$ be bounded by $[-\boldsymbol{w}_\delta, \boldsymbol{w}_\delta]$, where $\boldsymbol{w}_\delta$ is obtained by truncating the distribution $\mathcal{W}$ using a confidence interval (e.g., $\delta = 0.99$):

$$\therefore \boldsymbol{x}_{actual} \in [\boldsymbol{x} - \boldsymbol{w}_\delta, \boldsymbol{x} + \boldsymbol{w}_\delta]. \tag{3.12}$$

Therefore, the likelihood now becomes

$$P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h)) = \\ P(\boldsymbol{w} \in [min(\boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h))), max(\boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h)))]) \tag{3.13}$$

and in the case of constant velocity Dubins' car model, the likelihood can be written as:

$$\begin{aligned} &P(\boldsymbol{x}(t)|\boldsymbol{u}(t-h)) \\ &= P(\boldsymbol{w} \in [min(\boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h))), max(\boldsymbol{x}(t) - f(\boldsymbol{x}(t-h), \boldsymbol{u}(t-h)))]) \\ &= P(\boldsymbol{w} \in [min(\theta(t) - \theta(t-h) - \boldsymbol{u}(t-h)h)), max(\theta(t) - \theta(t-h) - \boldsymbol{u}(t-h)h))]) \\ &= P(\boldsymbol{w} \in [(\theta(t) - \boldsymbol{w}_\delta - (\theta(t-h) + \boldsymbol{w}_\delta) - \boldsymbol{u}(t-h)h)) \\ &\qquad , (\theta(t) + \boldsymbol{w}_\delta - (\theta(t-h) - \boldsymbol{w}_\delta) - \boldsymbol{u}(t-h)h))]) \\ &= P(\boldsymbol{w} \in [(\theta(t) - \theta(t-h) - \boldsymbol{u}(t-h) - 2\boldsymbol{w}_\delta), (\theta(t) - \theta(t-h) - \boldsymbol{u}(t-h) + 2\boldsymbol{w}_\delta)]) \end{aligned} \tag{3.14}$$

where the distribution of $\boldsymbol{w}$ is known so the above probability can be computed.

Update posterior at $(t-h)$ to prior at t

Recall we assumed that the inputs change by a an incremental amount at each step. To align with this assumption, we use the ϵ-transition approach to update our posterior distribution at the next time step. We also assume an original prior probability distribution $p_0(\boldsymbol{u})$ is defined, which is independent of any measurement. This is typically the uniform distribution over the chosen grid points. Then, we update the posterior to the new prior using

$$P(\boldsymbol{u}(t)) = (1-\epsilon)P(\boldsymbol{u}(t-h)|\boldsymbol{x}(t)) + \epsilon p_0(\boldsymbol{u}) \tag{3.15}$$

where $\epsilon \in [0, 1]$. This update rule indicates that there is a certain small part of the next prior distribution is sampled from the original prior distribution chosen at the begining of the simulation.

3.3.2 Reachability Analysis

Reachability analysis, given the knowledge of dynamic model and constraints, allow us to capture all the possible ways a system is allowed to evolve over time. This is particularly useful for our problem as this alone can already help ruling out the infeasible reappearance states for a particular disappearance state.

Optimal Control to Reachability

Conducting reachability analysis ultimately amounts to solving an optimal control problem. We briefly describes how one can formulate the optimal control problem into a reachability problem, which can be solved using the level set approach [1] and numerical scheme [6].

To compute the FRS, we first define the set of initial states as implicit surface function $l(x)$. This can be done using sign-distance functions or a combination of sign-distance functions. By setting this function as our final state cost and setting the running cost as zero, we can obtain the forward reach set, formally defined as $R(T) = \{x_t : \exists u, s.t.\ \dot{x} = f(x,u), x(0) \in L, x(s) = x_t, \forall s \leq T\}$ by solving maximization problem

$$\begin{aligned} \max_{u} \quad & J(x,t) = l(x(T)) \\ \text{s.t.} \quad & \dot{x} = f(x,u,t). \end{aligned} \tag{3.16}$$

This maximization problem can be written as solving a initial value PDE with the corresponding boundary condition,

$$\begin{aligned} & \tfrac{dV}{dt} + \max_{u}\{\Delta V(x(t),t) \cdot f(x,u)\} = 0 \\ & V(x(T),T) = l(x(T)). \end{aligned} \tag{3.17}$$

Using dynamic programming principle it can be shown that the value function $V(x,t)$ is the viscosity solution for the (3.17). This viscosity solution can be solved using the level set method and corresponding numerical scheme [6]. If we define the initial set as a zero-

sublevel of the Lipschitz function $l(x)$, the FRS after time T is the zero sublevel set of the value function.

For the above setup, the optimal control u^* is obtained using the Minimum (Maximum) principle by

$$H^* = \max_u \{\Delta V(x(t), t) \cdot f(x, u)\} \geq H = \Delta V(x(t), t) \cdot f(x, u)\}. \tag{3.18}$$

The result shows that for a Dubins car model, the control is bang-bang. In particular

$$u^* = u_{max} * sign(\frac{dV}{d\theta}). \tag{3.19}$$

The optimal u is used to compute the optimal Hamiltonian numerically, which is crucial for the computation of the value function in the level set scheme.

Set-valued reachability via level set methods

From the Bayesian method we obtain the probability for each discrete $\boldsymbol{u}_{\mathrm{i}} \in \boldsymbol{U}$ at t_0 (time of disappearance). We then generate a forward reach set for each $\boldsymbol{u}_{\mathrm{i}}$, from the disappear state. The idea here is based on set-value reachability. Essentially we use the system's forward model to predict future states and inputs; instead of using the stochastic disturbance $\boldsymbol{w}$ directly, the forward model uses set-valued to predict future states. Note that since we have measurement error for all (x, y,θ) states, our initial state becomes an interval where its width equals the measurement uncertainty. In 3 dimensional space, the initial set is a cuboid. There is one more issue we must solve in order to compute the reach set for each $\boldsymbol{u}_{\mathrm{i}}$. Due to the ϵ-transition rule for input u, the range will grow with time even when we pick a single value $\boldsymbol{u}_{\mathrm{i}}$ at t_0. Fortunately, the control will remain bang-bang and we can simply update the range at certain time intervals. We note that this will affects the computation of optimal Hamiltonian as the control used is different.

Querying the out-state

Since computing the aforementioned sets are computational expensive, we compute them prior to the experiment in an offline manner. With the pre-computed reachable sets, we can check if a given out-state of the target lies within the sets and determine whether it may belong to an in-state. Since these reach sets are computed in a relative coordinates and are invariant to rotation and translation, one can easily check if the relative state between disappearance and reappearance lies are contained by the sets. Fig. 3.2 shows 15 reach sets R_{i} corresponding to an input set discretized into 15 points. The black contours are just 2D projections of the 3D sets and will not be actually used for our query process. Red asteroid is the out-state being queried. Green dashed lines represent the 2-D slices of the forward sets R_{i} that contains the out-state at that particular theta. We note that just because a state lies within a set R_{i}'s 2D projection (black), it does not mean such state is contained within the set as other states (e.g. theta) must also be considered. For instance, Fig. 3.3 shows how an out-state (red asteroid) is in the 2D projection (black contour) of the reach set, but in reality it sits outside the 3-D surface at $\theta = 2.65$, as shown by the pink slice.

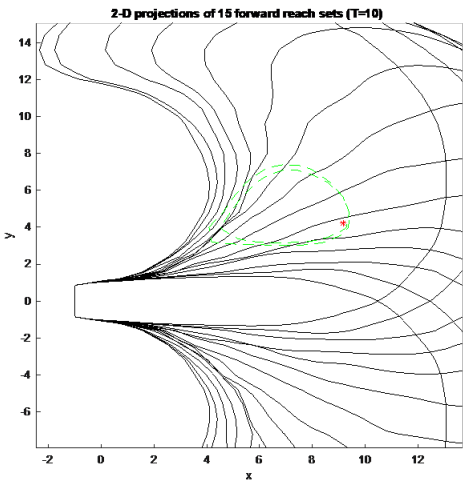


Figure 3.2. 2D projection of 15 forward reach sets ($T = 10$)

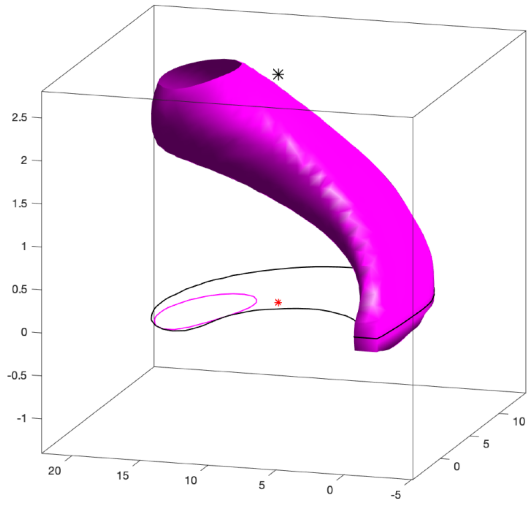


Figure 3.3. 3D reach set with 2D projections

3.3.3 Integrating Reachability with Bayesian Inference Results

Computing the Posterior Probabilities

Let, $\mathcal{X}_N(\boldsymbol{u})$ be the FRS after N steps with given initial state and control inputs $\boldsymbol{u}$. The initial state $\boldsymbol{x}(t_0)$ is known. Perform reachability analysis for every possible control input $\boldsymbol{u}_{\mathrm{i}}$ and get reach-sets $\mathcal{X}_N(\boldsymbol{u}_{\mathrm{i}})$, where: $\mathrm{i} = \{1, 2, \ldots, K-1\mathrm{i}\}$. $\mathcal{X}_N(\boldsymbol{u}_{\mathrm{i}})$ can be an interval, zonotope, ellipsoid or a convex polyhedron, depending on the approach used. The reach-sets are used to estimate reachable states. Then, the reachable states are used to compute the posterior probability with $\boldsymbol{x}_{out} \in \boldsymbol{x}(t_0 + Nh)$ for some target set $\mathcal{X}_u$ of interest. The posterior probabilities are derived by

$$P(\boldsymbol{x}_{out} \in \boldsymbol{x}(t_0 + Nh)|\boldsymbol{x}(t_0), \ldots, \boldsymbol{x}(0)) := \sum_{\mathrm{j} \in J} p_{\mathrm{i}} \tag{3.20}$$

where J is the set of indices $J = \{\mathrm{j} \in \{0, 1, 2, \ldots, K-1\}$ and defined by

$$J = \{\mathrm{j} \in \{0, 1, 2, \ldots, K-1\} | \mathcal{X}_N(\boldsymbol{u}_{\mathrm{j}}) \cap \boldsymbol{x}_{out} \neq \phi\} \tag{3.21}$$

Constructing Identity Probability Matrix

This approach gives the probability of identity of the exiting objects. The result can be put in the from of a matrix. For example, in the case of 3 objects, the matrix is written as

$$I = \begin{bmatrix} P(Out_1 = In_1) & P(Out_1 = In_2) & P(Out_1 = In_3) \\ P(Out_2 = In_1) & P(Out_2 = In_2) & P(Out_2 = In_3) \\ P(Out_3 = In_1) & P(Out_3 = In_2) & P(Out_3 = In_3) \end{bmatrix} \tag{3.22}$$

, which can be normalized in column direction to make it stochastic in the column direction.

$$\begin{aligned} ID(\mathrm{i}, \mathrm{j}) &= P(Out_{\mathrm{i}} = In_{\mathrm{j}} / Out_k, k \in \{1, 2, 3\}) \\ &= \frac{P(Out_{\mathrm{i}} = In_{\mathrm{j}})}{\sum_{k \in \{1,2,3\}} P(Out_k = In_{\mathrm{j}})} = \frac{I_{\mathrm{i,j}}}{\sum_{\mathrm{i}} I_{\mathrm{i,j}}} \end{aligned} \tag{3.23}$$

This idea is derived from the "Belief Matrix" from [67][67][68], which is defined as: "The belief matrix is a matrix B, in which elements b_{ij} represent the probability of object j having identity i. Belief matrix of the entire system is doubly-stochastic (i.e., each row sum and column sum is 1)". In our case the matrix can be stochastic in column direction alone.

3.4 Numerical Results

In simulation cases, the trajectories are generated by using (3.1) with control inputs u bounded in a range, which $u_{min} = -\boldsymbol{\pi}/9\ (rad/sec)$ and $u_{max} = \boldsymbol{\pi}/9\ (rad/sec)$. As u changes over time, the change rate is defined by $\epsilon[u_{min}, u_{max}]$ with ϵ equals to $1/8$. In order to generate noisy measurements, we add disturbance w to the trajectory with distribution $\mathcal{W}$ having mean equals to $[0, 0, 0]$ and standard deviation equals to $[0.01, 0.01, 0.001]$. With constant speed v equals to $2\ (m/sec)$ and $18\ (sec)$ of simulation time, we got 2 cases of trajectories. The first and second case demonstrates how the probabilities from Bayesian inference and reachability analysis can help us to obtain better result for identity association, respectively. In each cases we know each target measurement $\boldsymbol{x}(t)$ at each time step and the every time t_{in}, time at which occlusion starts, and t_{out}, time at which occlusion ends. Note that we only know the candidate disappear duration for the targets but we do not know which duration belongs to which target due to the identity issue yet to be solved.

Then proposed method is implemented into these cases and the following analysis are based on these results. u is divided in to 15 intervals for computing the priors and calculating reachability sets for the time t_0.

3.4.1 Scenario 1

In the first case, target 1 (blue) enters the occlusion area with a right-turning behavior while target 2 (red) enters straight from the left without much steering, as shown in Fig. 3.4. Target 3 has a mirrored behavior as target 1. We show this example to demonstrate how a more heavily biased distribution help generate the correct probability of identity.

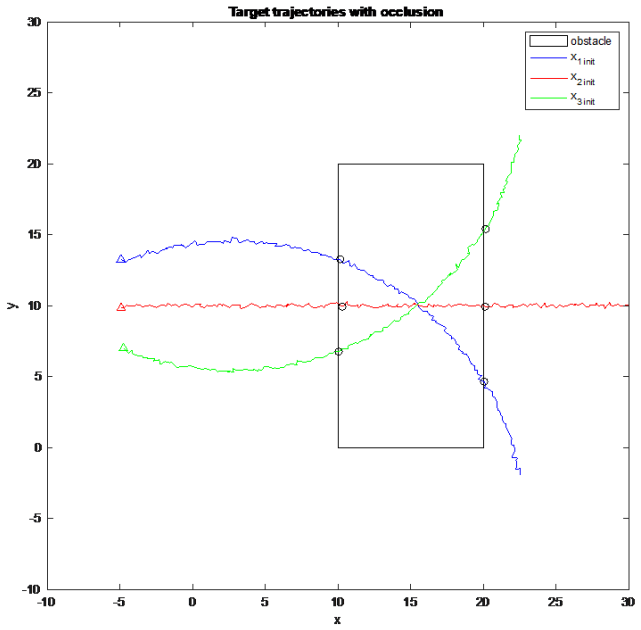


Figure 3.4. Target trajectory with occlusion (scenario 1)

Prior probability of Control Inputs

From the trajectory before t_0, we can compute the probability distribution of every control inputs $\boldsymbol{u}(t_0)$ for each target at t_0, right before the target disappears. The results is shown in Figure 3.5. Due to the continuously right-turn motion prior to t_0, target 1 has a higher probability in negative u. Target 3 represent a mirrored behavior as target 1 in this result. Obviously, target 2 keep moving one a straight line, so it has the highest probability on control input around $u = 0$.

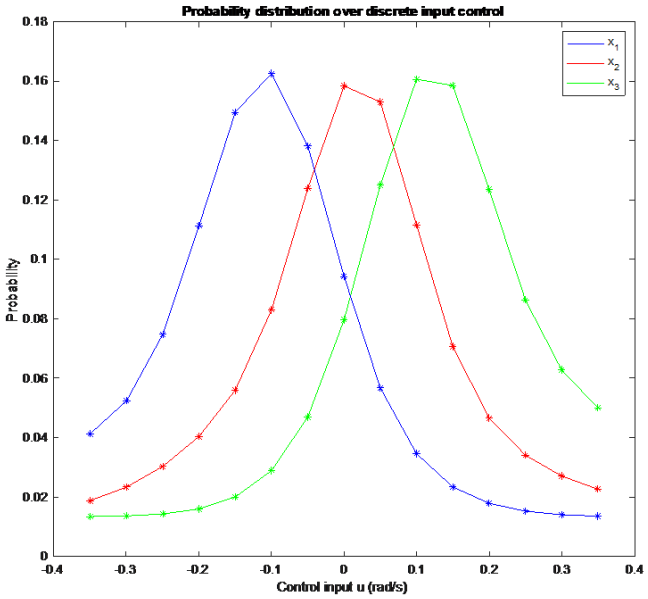
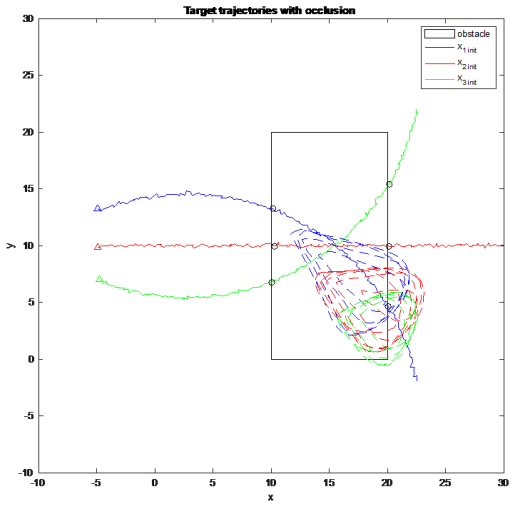


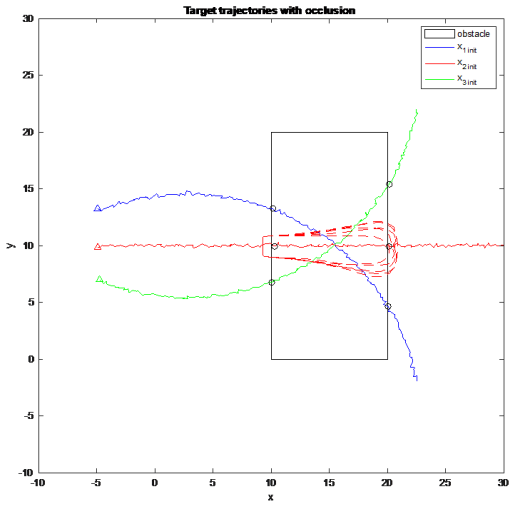
Figure 3.5. Probability distribution over discrete input control (scenario 1)

Reachability Results

Fig. 3.6a shows the reach sets (dashed lines) computed for the out-state (reappearing state) x_{out1} of target 1. The blue, red, green sets corresponds to the reach sets that contains the x_{out1} computed from all the possible in-states $x_1(t_0)$, $x_2(t_0)$, and $x_3(t_0)$ respectively. Since we ought to build a belief matrix of identity association purpose, we must conduct such look-up process for n^2 times where n is the number of targets. During each look-up, we slice the reachable set using the out-state's orientation and time of dissappearance. In Fig. 3.6a we can see there are four blue sets that contains x_{out1}, six red sets and 5 green sets that contains x_{out1}. This means we have to add up 4,6,5 probability values from the prior probability of u of x_1, x_2, and x_3. However, even though there are less sets containing x_{out1} from $x_{\text{in}1}$, the probabilities of the corresponding u's are higher, causing the probability of x_{out1} belonging to $x_{\text{in}1}$ to be the highest of all three. This precisely demonstrates how having an accurate prior distribution of u's is crucial to the identity association problem. Fig. 3.6b is a trivial case showing that x_{out2} can only be reached from $x_{\text{in}1}$, therefore assigning zero probability for the other two, as shown in Fig. 3.7.



(a) Reachibility sets of each $x(t_0)$ and $x_1(t_f)$ (scenario 1)



(b) Reachibility sets of each $x(t_0)$ and $x_2(t_f)$ (scenario 1)

Figure 3.6. Reachable Sets for Scenario 1

Identity Probability Matrices

Summing up the probability of each reachable set that can reach $x(t_f)$ from $x(t_0)$, we get the identity probability matrix in Fig. 3.7. The results verify the idea of proposed method. Target 1 has the probability of 0.8658, 0, and 0.1342 to emerged from the obstacle at $x_1(t_f)$, $x_2(t_f)$, and $x_3(t_f)$, respectively. Indeed, it has the highest probability to emerged from $x_1(t_f)$, which matches with the trajectory in Fig. 3.4. Target 2 and target 3 also have the similar results.

	in_1	in_2	in_3
out_1	0.5444	0.3571	0.0775
out_2	0	0.6298	0
out_3	0.0844	0.4430	0.5238

(a) Un-normalized

	in_1	in_2	in_3
out_1	0.8658	0.2497	0.1289
out_2	0	0.4404	0
out_3	0.1342	0.3098	0.8711

(b) Normalized

Figure 3.7. Identity probability matrix (scenario 1)

From above matrices the identity association result is:

$$Out_1 \rightarrow In_1 \qquad Out_2 \rightarrow In_2 \qquad Out_3 \rightarrow In_3$$

3.4.2 Scenario 2

In the second case, we show that our method can still provide a good result despite a less informative prior distribution on the control $\boldsymbol{u}_{\mathrm{i}}$'s at t_0. In cases where the probability of identities are similar for two or more in-states given an out-state, reachability can be used to eliminate unlikely options by looking at the result from other out-states.

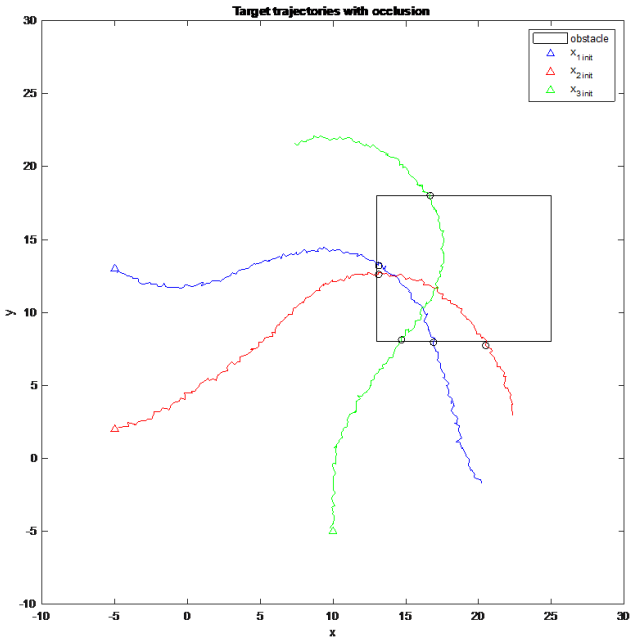


Figure 3.8. Target trajectories with occlusion (scenario 2)

Prior probability of Control Inputs

For this scenario, control input changes over time, so the trajectory is no longer keep turns in a single direction or move in a consistent straight line. From Fig. 3.8, Target 1 first does a period of left turn than change to right turn before it got occluded. Thus, as shown in Fig. 3.9, the peak of probability distribution of $\boldsymbol{u}$ locates near 0. We can also see that

target 3 has highest probability of continue making a right turn since this is what it had been doing prior to t_0.

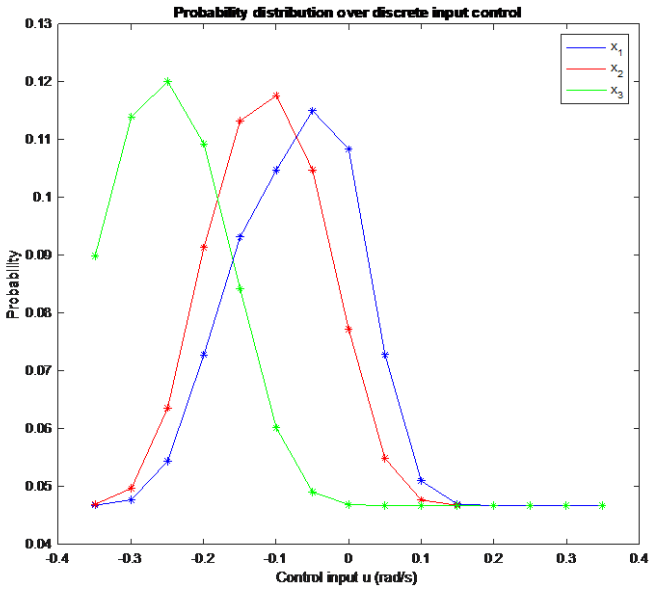
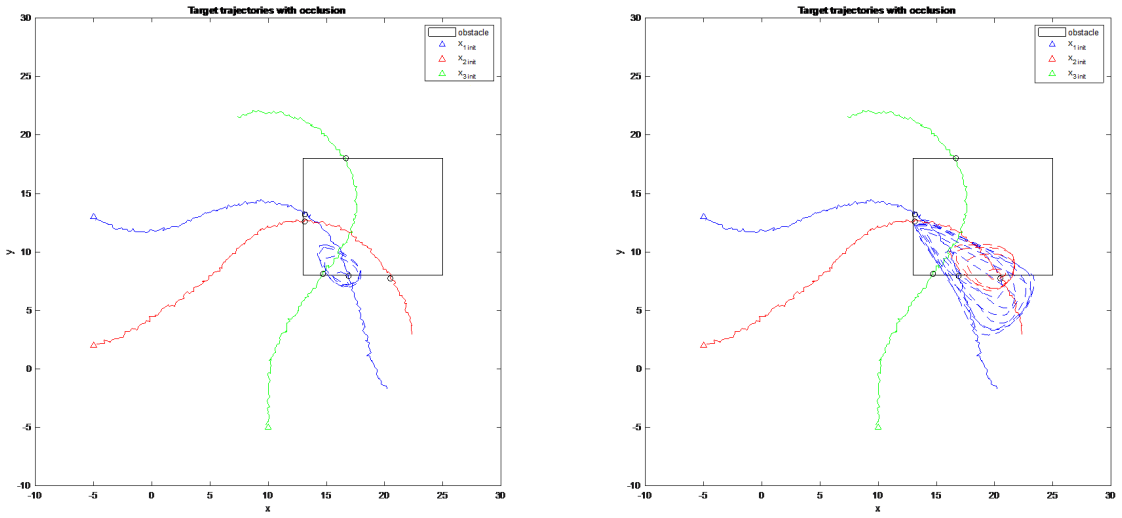


Figure 3.9. Probability distribution over discrete input control (scenario 2)

Reachibility Results

Again, we have overlaid the reach sets on the target trajectories in Fig. 3.6b and Fig. 3.10b. In Fig. 3.6b we again see a trivial case showing that x_{out1} can only be reached from x_{in1} given the exit orientation and exit time, so we can be very confident that x_{out1} belongs to x_{in1}. The more interesting case is shown in Fig. 3.10b where there are again more sets that can reach x_{out2} from x_{in1} than from x_{in2}. This time, however, the prior probability distribution for control is very similar for x_1 and x_2, causing the identity probability of x_{out2} belonging to x_{in1} to be actually higher. By looking at this result solely, one might misidentify target 2 to be target 1. However, if we look at the first row of identity matrix in Fig. 3.11. it is obvious that x_{in1} must lead to x_{out1}. Therefore, by elimination we can know that x_{out2} actually belongs to x_{in2}. This example demonstrates that our method can help identify targets even under a less accurate (or noisy) prior distribution of control.



(a) Reachibility sets of each $x(t_0)$ and $x_1(t_f)$ (scenario 2)

(b) Reachibility sets of each $x(t_0)$ and $x_2(t_f)$ (scenario 2)

Figure 3.10. Reachable Sets for Scenario 2

Identity Probability Matrices

According to the result in previous part, we sum up all the probabilities and get the identity probability matrices shown in Fig. 3.11. For the in-states of target 2 and target 3, both of them are assigned to the correct emerged point $x_2(t_f)$ and $x_3(t_f)$, respectively. However, for target 1, it has the higher probability, which is 0.7202, to emerged from occlusion at $x_2(t_f)$ rather than $x_1(t_f)$. This is because the measurement $x_1(t_0)$ of target 1 has an angle ψ that is more likely to direct to $x_2(t_f)$. However, this problem can be solved through elimination as mentioned above. In future work, we consider incorporating backward reachable sets in our proposed method, and we believe this ambiguous situation will be resolved more elegantly.

	in_1	in_2	in_3
out_1	0.2202	0	0
out_2	0.5667	0.2045	0
out_3	0	0	0.1400

(a) Un-normalized

	in_1	in_2	in_3
out_1	0.2798	0	0
out_2	0.7202	1	0
out_3	0	0	1

(b) Normalized

Figure 3.11. Identity probability matrix (scenario 2)

From above matrices the identity association result is:

$$Out_1 \rightarrow In_1 \qquad Out_2 \rightarrow In_2 \qquad Out_3 \rightarrow In_3$$

3.5 Conclusion

The proposed method computes an upper bound on the probability of exited objects being a particular target before occlusion using the reach-sets and Bayesian estimation of prior probabilities of control input. We have implemented the method in MATLAB using "Level Set Tool Box" for reach set computation. Two cases were simulated and the corresponding results were discussed. In the first case, we show how the probability distribution on the control input can help us identify targets even when their exit location is more reachable by other in-states. The second scenario demonstrates how our method can provide useful result under situations where the prior distributed cannot be fully trusted. The identity probability matrices for the 2 scenarios are obtained and the most probable identity of the trajectory is in agreement with the actual ground truth.

3.5.1 Limitations of the Method

The results of above method are sub-optimal when the variance of the noise is comparable to the change in state $\boldsymbol{x}$ caused by $\boldsymbol{u}$ in a time-step. The method heavily depends on the boundedness in the evolution of control inputs (3.5) with time which might not always be guaranteed. We also note our assumption that the target will continue to maneuver in a way similar to that exhibited before occlusion is a strong assumption and may not be practical.

3.5.2 Possible Extensions

The above problem of finding probabilities of grid points of u can be posed as clustering problem with discrete u's as clusters following a discrete distribution and the parameters of the distribution can be estimated using clustering algorithms like Expectation Maximization (EM). Another direction of future work is that the above simulations using FRSs can be augment with backward reachable set in similar fashion. This will potentially provide better results as information after reappearance is also used.

4. LEARNING CONTROL BARRIER FUNCTIONS WITH POLYTOPIC APPROXIMATION

4.1 Introduction

Safety is one of the most crucial factors that must be addressed before deploying an autonomous system into the real world. It is therefore necessary to integrate safety criteria into controller design process to ensure no undesired events take place. For instance, safety requirement must be enforced strictly for self-driving cars in order to keep humans unharmed. Moreover, damages to the system itself as well as the environment should be avoided. For the above reasons, designing a controller for safety-critical autonomous systems is a nontrivial task.

Recently, control barrier functions (CBF) become a popular choice to enforce safety on autonomous systems. Barrier function were originally developed to address the safety requirements of a system [69][70][71]. This idea was extended to consider controls and the notion of "control barrier function" was formally defined in [28]. Recently the work in [72][73] redefined the definition of *control barrier functions (CBF)* by extending Nagumo's theorem to the entire safe set (rather than just on the boundary). An important consequence is that one can now construct safe controller using such CBF since it is now defined at all points within the set. This is typically done through a *min-norm* controller that alters the nominal control in a minimally invasive manner [73]. In [74][75][76], CBFs are applied to develop safe controller for bipedal robots and enable them to walk on stepping stones. In automotive applications, CBF has be used to provide guarantees on safety features such as lane-keeping and adaptive cruise control [72][77][78]. Safety-critical controllers for aerial system such as quadrotors have also been synthesize using CBFs [79][80]. Experimental work has been done to prove the effectiveness of using CBF to control non-statistically stable systems such as a Segway in [81]. The application of CBF also extends into multi-robot systems [82][83][84][85], where safe maneuvers and methods for motion coordination are developed.

It is evident that control barrier functions has a wide range of application and its ability to provide provable safety guarantee is desirable. However, it is not clear how one can find

such CBF and its corresponding safe set for an arbitrary control system. This can, in general, be a difficult problem since CBF is very application-dependent.

Without knowing the algebraic form of the CBF, one cannot leverage the safety guarantee that provided by the CBF. There are several recent works investigated this problem via learning. In [86], a CBF is incrementally learned to cope with unmodel disturbance, with a conservative CBF provided at the beginning of the learning process. Authors of [87] introduce a neural-network-based approach to jointly learn the control policy along with CBF and Lyapunov function. This approach requires prior knowledge of safe and unsafe sets. [88] proposed an optimization based approach to learning CBF from expert demonstrations. The paper also proves the validity of the learned CBF under suitable assumptions of smoothness on the underlying dynamics and the learned CBF, and under sufficiently fine sampling. In [89], the constraints of the workspace, i.e. the CBFs, are parameterized by linear functions. The proposed approach incrementally updates the linear functions from human demonstrations. In [90], with given safe and unsafe data, support vector machine is used to learn a CBF. These aforementioned methods however generally neglect the control constraints, and therefore the condition that requires the time derivative of CBF be negative semi-definite can almost always be satisfied. In reality, this can be a problem since there are actuator constraints and input bounds that limit the system. For this reason, one must make an effort to ensure the input constraints do not conflict with the required conditon on the time derivative of CBF. The authors of [91] consider the input constraint, however, they assume a given control law, which is somewhat limited.

In [87], [90], the proposed methods rely on given classified data from safe set and unsafe set. This is impractical as it is hard to select data in a set, i.e. the safe set, that has the property of forward invariant. In this paper, we will select data from an allowable set defined by some safety measure $\rho(\boldsymbol{x})$. Such safety measure is tailored for different applications. The allowable set is not necessarily forward invariant, therefore, in the context of CBF, it is not safe. However, it is easy to generate, and the safe set defined by CBF will be a subset of the allowable set.

The main contribution of this work is an optimization framework for learning control barrier functions from application specific safety measures $\rho(x)$. We parameterize the learned control barrier function using a collection of linear inequalities and the corresponding safe invariant set is represented as the intersection of half spaces. We provide validation for the learned control barrier function and perform a series of simulations to show the effectiveness of our method.

4.2 Preliminary and Problem Formulations

4.2.1 System Dynamics

We consider a nonlinear affine control system of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}, \tag{4.1}$$

where $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ denotes the system states, with $\mathcal{X}$ as the system state space. The control input $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^m$ is bounded by admissible set of input and here, we consider $\mathcal{U} = \{\boldsymbol{u} | \boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}\}$, where $\leq$ denotes entry-wise inequality. Let $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ be locally Lipschitz functions, and denote $I(\boldsymbol{x}_0) = [0, \tau_{max})$ as the maximum interval of existence on which $\boldsymbol{x}(t)$ is a unique solution to (4.1) from the inital condtion $\boldsymbol{x}(0) = \boldsymbol{x}_0 \in \mathcal{X}$.

4.2.2 Safety and Control Barrier Function

At a high level, our goal is to search for a control barrier function that can certify the safety of a system. In fact, the existence of control barrier function implies the safety of the control system [73]. To do so, we must first define the notion of *safety* for a system. In general, safety can be expressed in terms of set invariance.

Definition 4.2.1 (Invariant set [92])**.** *A set $\mathcal{C}$ is forward-invariant if for any initial state $\boldsymbol{x}_0 \in \mathcal{C}$, $\boldsymbol{x}(t) \in \mathcal{C}$ $\forall t \in I(x_0)$. Then, the system (4.1) is* safe *w.r.t. the set $\mathcal{C}$ if $\mathcal{C}$ is forward-invariant [92].*

Simply put, if a system that start in a safe set $\mathcal{C}$ always remains within $\mathcal{C}$, then the system is considered *safe*. We can now formally state the definition of a control barrier function:

Definition 4.2.2 (Control barrier function [92])**.** *Consider the control system (4.1), a set of safe states $\mathcal{X}_s \subseteq \mathcal{X}$, and a set of unsafe states $\mathcal{X}_u = \mathcal{X} \setminus \mathcal{X}_s$. A continuous differentiable function $h(x)$ is a* control barrier function (CBF) *if there exist an extended class $\mathcal{K}_\infty$ function α such that*

$$h(\boldsymbol{x}) \leq 0 \quad \forall \boldsymbol{x} \in \mathcal{X}_s \tag{4.2}$$

$$h(\boldsymbol{x}) > 0 \quad \forall \boldsymbol{x} \in \mathcal{X}_u \tag{4.3}$$

$$L_f h(\boldsymbol{x}) + L_g h(\boldsymbol{x}) u \leq -\alpha(h(\boldsymbol{x})) \quad \forall \boldsymbol{x} \in \mathcal{X}. \tag{4.4}$$

Note $L_f h(\boldsymbol{x})$ and $L_g h(\boldsymbol{x})$ are Lie-derivatives of $h(\boldsymbol{x})$, e.g., $L_f h(\boldsymbol{x}) = \frac{\partial h(\boldsymbol{x})}{\partial \boldsymbol{x}} f(\boldsymbol{x})$ and $L_g h(\boldsymbol{x}) = \frac{\partial h(\boldsymbol{x})}{\partial \boldsymbol{x}} g(\boldsymbol{x})$. We also assume that $\frac{\partial h(x)}{\partial x} \neq 0$. We highlight that condition (4.4) is different from the that stated in Nagumo's theorem [69] (which essentially states that $\dot{h}(\boldsymbol{x}) \leq 0$ on $\partial \mathcal{X}_s$ is a necessary and sufficient condition for set invariance), but under mild conditions on the set $\mathcal{X}_s$, it is proven that conditions (4.2-4.4) are also necessary and sufficient for forward invariance [73] of $\mathcal{X}_s$ and hence guaranteeing the safety of the system (4.1).

4.2.3 Finding CBF for Input-Constrained Systems

Assumption 1. *There exists a set of allowable states $\mathcal{X}_\rho = \{\boldsymbol{x} \in \mathcal{X} \mid \rho(\boldsymbol{x}) \leq 0\}$, which is defined as the zero-sublevel set of some known safety measure $\rho : \mathcal{X} \to \mathbb{R}$.*

This is not a strong assumption since such safety measure is usually user-specified or can be relatively easily obtained by inspecting the application.

Due to the input constraints considered, not all points in $\mathcal{X}_\rho$ can always be rendered safe and therefore one cannot say ρ is a control barrier function, that is, $\mathcal{X}_\rho$ is not equivalent to $\mathcal{X}_s$. For example, if $\boldsymbol{x} \in \partial \mathcal{X}_\rho$, there exist no $\boldsymbol{u}$ such that $\dot{h}(\boldsymbol{x}) \leq 0$ due to the finite

control limit. Specifically, the introduction of input constraints may cause certain points in the allowable set $\mathcal{X}_\rho$ to lose property of forward invariance, e.g., the input that is required to keep the system in the safe may lie outside of the input constraint and lead to an infeasible problem. This motivates us to find a function such that all the states within its zero-sublevel set can be rendered forward invariance with the given input constraint and is a strict subset of the geometric safe set, i.e. satisfying Definition 4.2.2. Once such function is found, we can claim such function to be a valid control barrier function and the safe-invariant set is indeed the zero-sublevel set of this function. We now formally state our problem statement.

Our **problem of interest** is to find a control barrier function $h(x)$ with the consideration of control input constraints, such that

- The safe set $\mathcal{X}_s$ is a strict subset of the set of the allowable states $\mathcal{X}_\rho$, i.e. $\mathcal{X}_s = \{x \mid h(\boldsymbol{x}) < 0\} \subset \mathcal{X}_\rho$.
- $\mathcal{X}_s$ is the safe set with a substantially large volume[1].

Remark 1. *Note that it is important to distinguish between the* allowable set *defined by safety measure $\rho(x)$ and the* safe set *defined by control barrier function. These two sets are not equivalent in all cases but the* safe set *is always a subset of the set set, e.g., $\mathcal{X}_s \subset \mathcal{X}_\rho$.*

4.3 Methodology

To make this problem tractable and to efficiently search for a valid control barrier function, we parameterize such function using polytopic approximation similar to [89]. In particular, a collection of linear inequalities will be used to denote the control barrier function and resulting safe set is represented as the intersection of safe half spaces. Safe half spaces are defined by linear inequalities $h_{\mathrm{i}}(\boldsymbol{x}) = \boldsymbol{a}_{\mathrm{i}}^T \boldsymbol{x} + b_{\mathrm{i}} \leq 0$, with $\boldsymbol{a}_{\mathrm{i}} \in \mathbb{R}^n$, $b_{\mathrm{i}} \in \mathbb{R}$; $\mathrm{i} = \{1, \cdots, L\}$, where L is the number of linear inequalities that are used to construct the safe set. The CBF defined by the collection of linear inequalities can then be expressed by

$$h_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}, \tag{4.5}$$

[1]↑we do not claim the learned safe set has maximum volume, but we attempt to create a similar effect.

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \vdots \\ \boldsymbol{a}_L^T \end{bmatrix} \in \mathbb{R}^{L\times n}, \qquad \boldsymbol{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_L \end{bmatrix} \in \mathbb{R}^{L}, \tag{4.6}$$

Here, the CBF should possess the same properties as mentioned in (4.2-4.4), although we note that the dimensions are different, where $h_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x}) \in \mathbb{R}^L$.

To ensure the learned function is a valid CBF, it must satisfy properties stated in (4.2-4.4). The condition (4.4) requires $\dot{h}_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x}) \leq -\alpha(h(\boldsymbol{x})), \forall \boldsymbol{x} \in \mathcal{X}$, meaning there is always a feasible input that allows us to drive the system towards the interior of zero-sublevel of $h_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x})$. In our case, we relax such condition and only require that $\dot{h}_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x}) \leq -\alpha(h(\boldsymbol{x})), \forall \boldsymbol{x} \in \mathcal{X}_\rho$

We note that this not only ensures the invariance property of $\mathcal{X}_s$ but also makes it asymptotically stable in $\mathcal{X}_\rho$. To achieve this, we propose a sampling-based optimization approach. M and N are number of sample points generated from the interior of $\mathcal{X}_\rho$ and the boundary of the set, respectively. The optimization problem is set up as below:

$$\begin{aligned} \min_{\boldsymbol{A},\boldsymbol{B}} \quad & \sum_{\mathrm{j}=1}^{N} \max_{\boldsymbol{a}_\mathrm{i},b_\mathrm{i}} (\boldsymbol{a}_\mathrm{i}^T \boldsymbol{x}_\mathrm{j} + b_\mathrm{i}), \\ \text{s.t.} \quad & \min_{\boldsymbol{u}} (\boldsymbol{a}_\mathrm{i}^T \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{a}_\mathrm{i}^T \boldsymbol{g}(\boldsymbol{x}_k)\boldsymbol{u} + \alpha(\boldsymbol{a}_\mathrm{i}^T \boldsymbol{x}_k + b_\mathrm{i})) \leq 0, \\ & \max_{\boldsymbol{a}_\mathrm{i},\boldsymbol{b}_\mathrm{i}} (\boldsymbol{a}_\mathrm{i}^T \boldsymbol{x}_\mathrm{j} + b_\mathrm{i}) \geq 0, \end{aligned} \tag{4.7}$$

The objective of this optimization problem is designed to create an effect similar to that of maximizing the volume of the safe set, which is the zero sublevel set of the learned CBF. $\boldsymbol{x}_\mathrm{j}$ denotes the sample points on the boundary of the set of allowable states, where $\boldsymbol{x}_\mathrm{j} \in \{\boldsymbol{x} | \rho(\boldsymbol{x}) = 0\}, \mathrm{j} = \{1, ..., N\}$. To enlarge the learned safe set, we want to minimize the distance between each boundary sample $\boldsymbol{x}_\mathrm{j}$ and the 'line' $\boldsymbol{a}_\mathrm{i}^T \boldsymbol{x}_\mathrm{j} + b_\mathrm{i} = 0$ that is closest to $\boldsymbol{x}_\mathrm{j}$, for $\mathrm{j} = \{1, ..., N\}$. If we limit the norm of the coefficients $\boldsymbol{a}_\mathrm{i}$ to be one, then the linear equations $(\boldsymbol{a}_\mathrm{i}^T \boldsymbol{x}_\mathrm{j} + b_\mathrm{i})$ are essentially signed distance functions. Thus, if we assume that for each point $\boldsymbol{x}_\mathrm{j}$ on the boundary, there exist at least one linear function to which the value of $\boldsymbol{x}_\mathrm{j}$

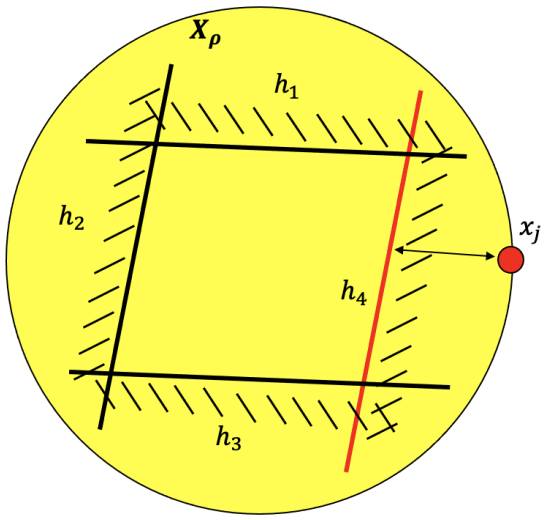


Figure 4.1. Geometric meaning of our objective function

is positive ($\boldsymbol{x}_\mathrm{j}$ lies 'outside' of this 'side' of the polytope), then $\max_{\boldsymbol{a}_\mathrm{i},\boldsymbol{b}_\mathrm{i}}(\boldsymbol{a}_\mathrm{i}^T\boldsymbol{x}_\mathrm{j}+b_\mathrm{i})$ corresponds to such linear function. By minimizing the distance between the boundary sample and this 'edge', the polytope is stretched as much as possible. Fig. 4.1 illustrates this idea. The figure shows four linear functions and the shaded regions represent the positive half-spaces w.r.t. each $h_\mathrm{i}(\boldsymbol{x})$. Given a boundary sample $\boldsymbol{x}_\mathrm{j}$ (red point), we can see that h_4 is the only linear function to which the value of x_j is positive, therefore $\max_{\boldsymbol{a}_\mathrm{i},\boldsymbol{b}_\mathrm{i}}(\boldsymbol{a}_\mathrm{i}^T\boldsymbol{x}_\mathrm{j}+b_\mathrm{i})$ refers to the distance between such 'edge' and the sample point. In short, we are essentially minimizing the the overall distance between the boundary of $\mathcal{X}_s$ and that of $\mathcal{X}_\rho$, therefore enlarging the volume of $\mathcal{X}_s$ since $\mathcal{X}_\rho$ is fixed.

The first constraint of the optimization problem is introduced in coherence with (4.4) to ensure the forward invariance of the learned safe set. We sample the interior of the set of allowable states and denote these interior samples as $\boldsymbol{x}_k = \{\boldsymbol{x}|\rho(\boldsymbol{x}) \leq 0\}$. Input constraint is considered, where $\boldsymbol{u} \in [\boldsymbol{u}_{min}, \boldsymbol{u}_{max}]$. For each sample point, the time derivative of function $h_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x})$ must be smaller than a Class-$\mathcal{K}_\infty$ function. To consider the bounded input, for each point, $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ are substituted into the constraint (separately for each linear function). The two resulting values from the constraint are compared, and if the smaller of the two is negative-definite then the constraint is satisfied. The purpose of this constraint is

to ensure the existence of an *admissible input* value that can render the safe set $\mathcal{X}_s$ forward invariant.

The second constraint corresponds to ensuring that the safe set learned $\mathcal{X}_s$ is a strict subset of the geometrical safe set $\mathcal{X}_\rho$, as shown in Fig. 4.2. Similar to the objective function, for each point on the boundary of $\mathcal{X}_\rho$, its value w.r.t the nearest linear function $h_{\mathrm{i}}(\boldsymbol{x})$ should be greater or equal to zero. Geometrically, this is equivalent to requiring for every boundary sample there exist at least one $h_{\mathrm{i}}(\boldsymbol{x})$ of which the sample lies on the positive half-space. Moreover, this constraint means that the boundary of $\mathcal{X}_\rho$ is not in the interior of the safe set $\mathcal{X}_s$. Thus, the learned $\mathcal{X}_s$ is always a strict subset of $\mathcal{X}_\rho$. Without this constraint, the zero-sublevel set of learned function $h_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{x})$ may extend beyond $\mathcal{X}_\rho$, therefore violating our requirements in section 4.2.3 and become what is depicted in Fig. 4.3.

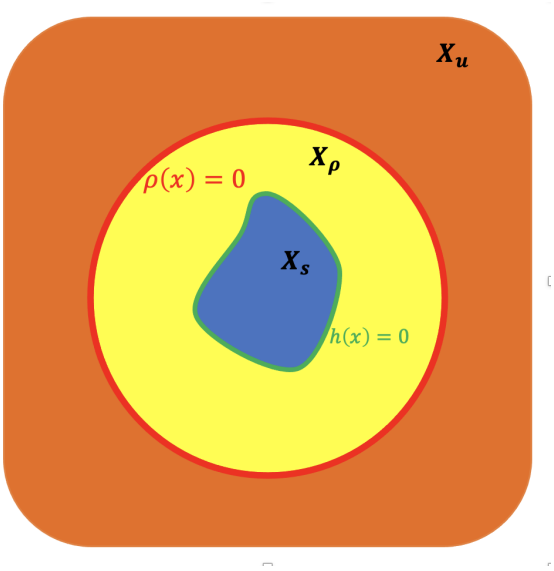


Figure 4.2. Illustration of CBF and the corresponding safe-invariant set

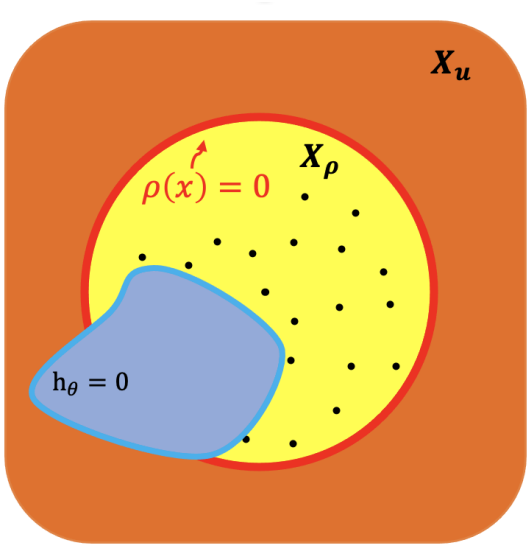


Figure 4.3. Learned safe-invariant set with missing Type II constraint

To obtain a desirable solution, we restrict the norm of the coefficients in $\boldsymbol{a}_\mathrm{i}$ to be one, as mentioned above. This is necessary for both ensuring the linear function h_i is a signed distance function and ensuring we do not get a trivial solution from the optimization. We note that this norm equality constraint makes the problem non-convex and therefore difficult to obtain a global minimum. To alleviate this, we feed in an initial condition that represent a convex polytope with L faces.

4.4 Numerical Results

4.4.1 Learning Control Barrier Function from Allowable Set

To demonstrate the effectiveness of our method discussed in section III, we apply it to a real system and show how the learned CBF can be used to keep the system from entering undesired states. We consider the nonlinear Moore-Greitzer jet engine model in no-stall mode used in [93] as our example. The dynamics of this system is given as

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} x_2 - \frac{3}{2}x_1^2 - \frac{1}{2}x_1^3 \\ x_1 \end{bmatrix} \text{ and } \boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix} 0 \\ -1 \end{bmatrix},$$

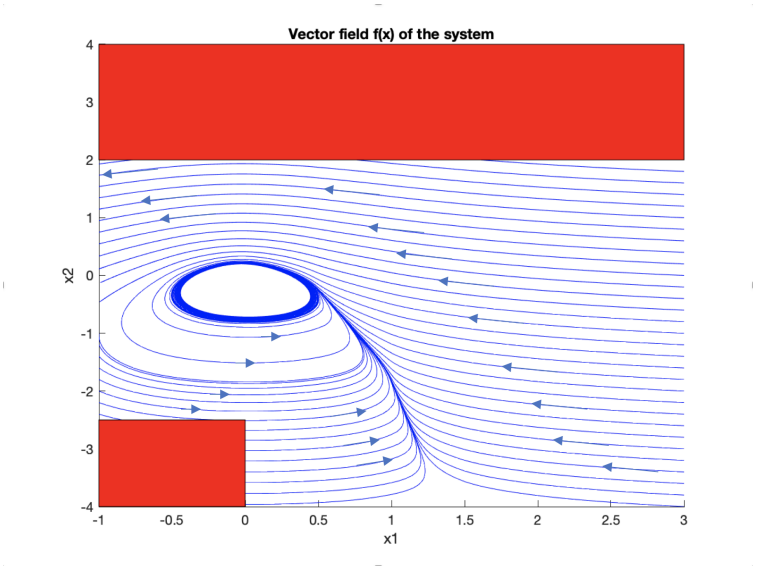


Figure 4.4. Vector field $f(x)$ of the system: X_u are represented by the red filled regions and the X_ρ is the unfilled region

where $\boldsymbol{x} = [x_1, x_2]^T$ represents our system states. Physically, $x_1 = 1-\Phi$ and $x_2 = \Psi-\psi-2$ are quantities proportional to the mass flow Φ and the pressure rise Ψ, respectively. Similar to [93], we consider the state-space of interest $X = [-1, 3] \times [-4, 4]$ and the unsafe region $X_u = [-1, 0] \times [-4, 2.5] \cup [-1, 3] \times [2, 4]$. The geometric safe set is then $X_\rho = X \setminus X_u$. The considered domain overlaid with the system's vector field is depicted in Fig. 4.4. Recall that our goal is to learn a control barrier function such that it's zero-sublevel set is a subset of X_ρ.

By applying the proposed optimization-based method, we are able to learn the control barrier function in the form of a collection of linear functions. For sampling, we took $N = 200$ samples from the interior of X_ρ and $M = 200$ samples on the boundary ∂X_ρ. We set the input constraint to be $U_{feasible} = [-9, 9]$ and we use $F = 6$ linear functions to perform our polytopic approximation. The optimization problem is solved using the CasADi solver [cite] that uses an interior point method. The linear functions learned are $h_1 = -0.75x_1 - 0.66x_2 - 1.68$, $h_2 = 0.95x_1 - 0.32x_2 - 2.36$, $h_3 = 0.981x_1 + 0.20x_2 - 1.15$, $h_4 = -0.28x_1 + 0.96x_2 - 1.26$,

$h_5 = -0.07x_1 - 0.007x_2 - 0.08$, and $h_6 = -0.20x_1 - 0.98x_2 - 2.87$. The corresponding safe-invariant set is the intersection of half spaces of these linear functions, e.g.,

$$X_{cbf} = \bigcap_{\mathrm{i}=1}^{6} \{x | h_{\mathrm{i}}(x) \leq 0\}.$$

4.4.2 Safe Control using Learned CBF

With the control barrier function obtained, we can now utilize the CBF-QP framework to implement a safe controller with constraints derived from the learned CBF. We first assume we have access to a nominal control input from a separate controller that does not guarantee system safety and can drive the system to dangerous states (X_u for instance). This controller can be a human controlling the system or an optimal controller that minimizes certain objectives. The goal is then to alter the nominal input in an minimally invasive way such that system safety is guaranteed [cite]. Below shows the CBF-QP framework

$$u(x) = \arg\min \ \frac{1}{2}\|\tilde{u}(x) - u\|^2 \tag{4.8}$$

$$s.t. \ \mathbf{A}f(x) + \mathbf{A}g(x)u \leq -\alpha(\mathbf{A}x + \mathbf{b}) \tag{4.9}$$

$$u_{min} \leq u \leq u_{max} \tag{4.10}$$

where $\mathbf{A}$ and $\mathbf{b}$ are as defined in 4.6.

The learned safe invariant set is shown in Fig. 4.5. The shaded region corresponds to the half-space $\{x | h_{\mathrm{i}}(x) \geq 0\}$ and the unshaded region is the invariant set X_{cbf}. We can see that X_{cbf} is a subset of X_ρ and that $X_{cbf} \cup X_u = \emptyset$. A nominal control $u_{nominal} = -3$ is applied to the system starting at $x_0 = [0.5, 1]$. The CBF-QP framework (4.8-4.10) is then used to obtain a safe control. The magenta line and the green line in Fig. 4.5 represents the system trajectories with and without the CBF-QP, respectively. We note that the nominal input drives the system into X_u since it does not concern about safety. On the other hand, the controller from CBF-QP kept the sytem within X_{cbf} via a smooth constrained trajectory,

hence rendering the set forward invariant. Fig. 4.6 denotes a similar case with system beginning at the initial state $x_0 = [-0.5, -1.5]$.

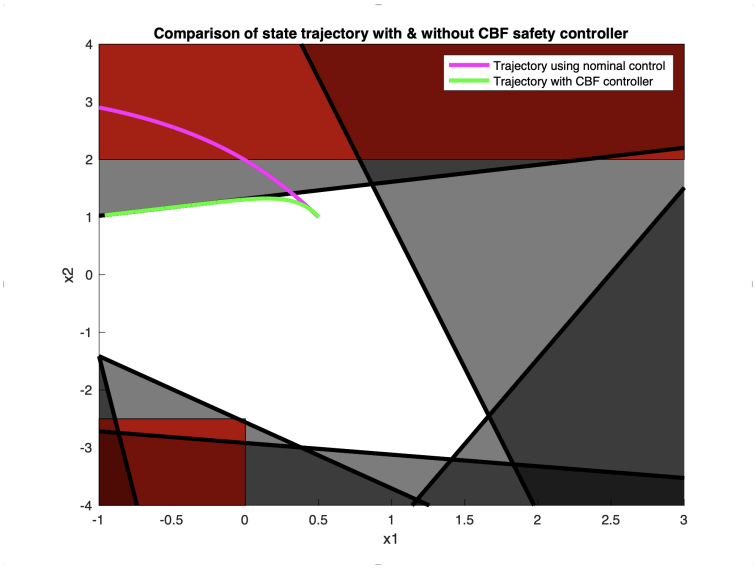


Figure 4.5. Comparison of state trajectory with and without CBF safe controller.

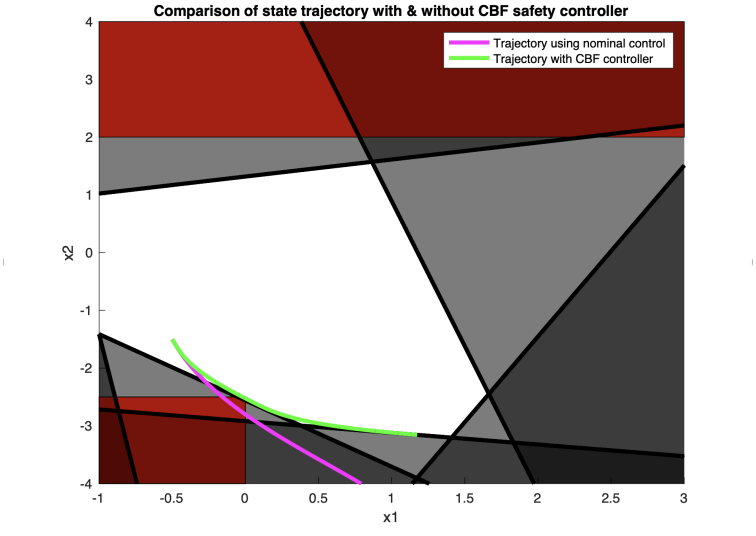


Figure 4.6. Comparison of state trajectory with and without CBF safe controller.

4.4.3 Effect of Input Constraints on Safe Set

Here, we elucidate how the bounds on control input of the system affects the control barrier function learned and the corresponding safe set. Fig. 4.7 and 4.8 show different safe sets learned using $L = 6$ linear functions and the input bounds $u_{\min/\max}$ equals to $\pm\infty$, ± 9, and 0, respectively. In Fig. 4.7, since the control is unbounded, this is equivalent to not enforcing constraint (4.9) in the optimization problem above. Recall that if the control is unbounded, we said the condition (4.4) can always be satisfied (if the system is control-affine) by selecting a large magnitude input. We note in Fig. 4.7 the learned safe set does not include the bottom left region and this can be attributed to the limited expressive power of the polytopic representation. Comparing this with Fig. 4.8, we can see the volume of the safe set in Fig. 4.8 is significantly smaller due to the control bounds. The shaded region in Fig. 4.8 can be interpreted as states that cannot be render forward invariant by any input within the given bounds. Intuitively, these may be states that are too close to the unsafe states and the input required to drive the system away from $\mathcal{X}_u$ is well beyond the system's physical limit.

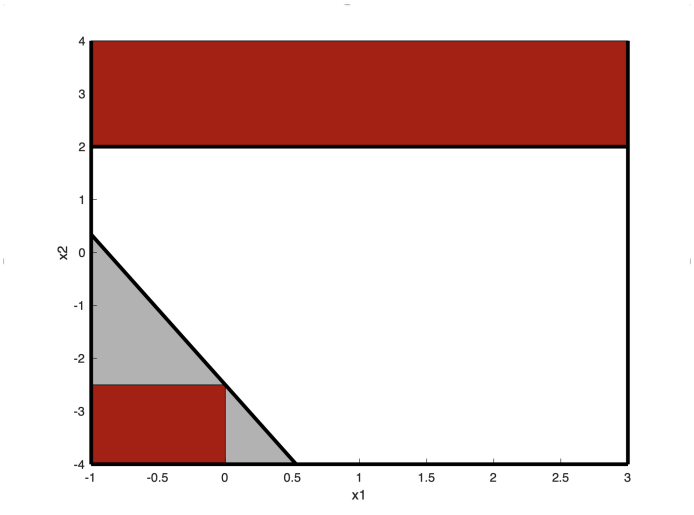


Figure 4.7. Safe set learned using $u_{min/max} = \pm\infty$

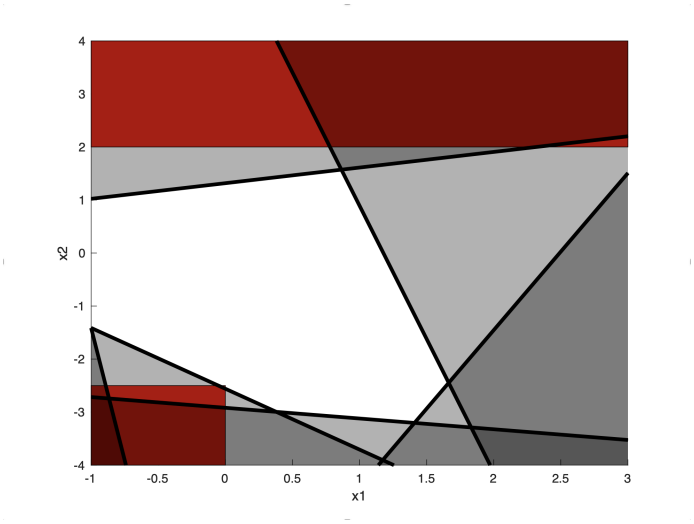


Figure 4.8. Safe set learned using $u_{min/max} = \pm 9$

4.5 Conclusion and Future Work

In this work, we propose a sampling-based optimization framework to learn a control barrier function from an application-specific safety measure. We show that given a nonlinear affine system with input constraints our method is able to learn a set of linear functions and that the invariant set is indeed the intersection of half-spaces of these functions. We demonstrate the validity of the approach through a min-norm safe controller in which the constraints are derived using the learned CBF. There are multiple directions that we would like to explore as future work. First, we wish to establish a formal proof that enable us to claim our objective function is indeed maximizing the volume of the learned safe set. Further, we would like to leverage the results in [88] and use Lipschitz properties to prove that, given the sampling is done at a fine enough resolution, if all states sampled satisfy our constraints, then even states that are not sampled will also satisfy such constraints. Another direction is to explore other representation such as sum-of-squares or ellipsoidal approximation in hope to improve the expressive power. Finally, we believe this work can be extended to a data-driven approach where we learn or correct the safe set in real-time as measurements are received.

REFERENCES

[1] I. M. Mitchell, "Application of level set methods to control and reachability problems in continuous and hybrid systems," Ph.D. dissertation, Stanford University, Jul. 2002.

[2] S. Kong, S. Gao, W. Chen, and E. Clarke, "Dreach: δ-reachability analysis for hybrid systems," Apr. 2015, pp. 200–205. DOI: 10.1007/978-3-662-46681-0_15.

[3] P.-J. Meyer, A. Devonport, and M. Arcak, "Tira," *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, Apr. 2019. DOI: 10.1145/3302504.3311808. [Online]. Available: http://dx.doi.org/10.1145/3302504.3311808.

[4] M. Althoff, "An introduction to cora 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.

[5] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, vol. 50, no. 7, pp. 947–957, 2005. DOI: 10.1109/TAC.2005.851439.

[6] I. M. Mitchell. (). "A toolbox of hamilton-jacobi solvers for analysis of nondeterministic continuous and hybrid systems." https://www.cs.ubc.ca/ mitchell/ToolboxLS/. (accessed: 05.01.2020).

[7] R. Isaacs, "Ifferential games: A mathematical theory with applications to warfare and pursuit, control and optimization," *Courier Corporation*, 1999.

[8] S. Bansal, M. Chen, S. L. Herbert, and C. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2242–2253, 2017.

[9] J. Ding, E. Li, H. Huang, and C. J. Tomlin, "Reachability-based synthesis of feedback policies for motion planning under bounded disturbances," pp. 2160–2165, 2011. DOI: 10.1109/ICRA.2011.5980268.

[10] J. H. Gillula, G. M. Hoffmann, H. Huang, M. P. Vitus, and C. J. Tomlin, "Applications of hybrid reachability analysis to robotic aerial vehicles," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 335–354, 2011. DOI: 10.1177/0278364910387173. eprint: https://doi.org/10.1177/0278364910387173. [Online]. Available: https://doi.org/10.1177/0278364910387173.

[11] Y. Zhou and J. S. Baras, "Reachable set approach to collision avoidance for uavs," pp. 5947–5952, 2015. DOI: 10.1109/CDC.2015.7403154.

[12] J. Ding, J. Sprinkle, S. S. Sastry, and C. J. Tomlin, "Reachability calculations for automated aerial refueling," pp. 3706–3712, 2008. DOI: 10.1109/CDC.2008.4738998.

[13] J. S. Jang and C. Tomlin, "Control strategies in multi-player pursuit and evasion game," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*. DOI: 10.2514/6.2005-6239. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2005-6239. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2005-6239.

[14] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 1517–1522. DOI: 10.1109/CDC.2017.8263867.

[15] C. Dabadie, S. Kaynama, and C. J. Tomlin, "A practical reachability-based collision avoidance algorithm for sampled-data systems: Application to ground robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 4161–4168. DOI: 10.1109/IROS.2014.6943149.

[16] M. Chen, Q. Hu, C. Mackin, J. F. Fisac, and C. J. Tomlin, "Safe platooning of unmanned aerial vehicles via reachability," pp. 4695–4701, 2015. DOI: 10.1109/CDC.2015.7402951.

[17] R. Takei, R. Tsai, H. Shen, and Y. Landa, "A practical path-planning algorithm for a vehicle with a constrained turning radius: A hamilton-jacobi approach," Jan. 2010.

[18] S. Bansal, A. Bajcsy, E. Ratner, A. D. Dragan, and C. J. Tomlin, *A hamilton-jacobi reachability-based framework for predicting and analyzing human motion for safe planning*, 2020. arXiv: 1910.13369 [cs.RO].

[19] A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, J. F. Fisac, S. Deglurkar, A. D. Dragan, and C. J. Tomlin, "A scalable framework for real-time multi-robot, multi-human collision avoidance," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 936–943. DOI: 10.1109/ICRA.2019.8794457.

[20] M. Chen, S. Herbert, and C. Tomlin, "Exact and efficient hamilton-jacobi guaranteed safety analysis via system decomposition," May 2017. DOI: 10.1109/ICRA.2017.7989015.

[21] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of reachable sets and tubes for a class of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 11, pp. 3675–3688, 2018. DOI: 10.1109/TAC.2018.2797194.

[22] I. Mitchell and C. Tomlin, "Overapproximating reachable sets by hamilton-jacobi projections," *Journal of Scientific Computing*, vol. 19, Nov. 2002. DOI: 10.1023/A:1025364227563.

[23] M. Chen, S. Herbert, and C. J. Tomlin, "Fast reachable set approximations via state decoupling disturbances," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 191–196. DOI: 10.1109/CDC.2016.7798268.

[24] I. Hwang, D. Stipanovic, and C. Tomlin, "Polytopic approximations of reachable sets applied to linear dynamic games and a class of nonlinear systems," *Systems and Control: Foundations and Applications*, pp. 3–19, Jan. 2005. DOI: 10.1007/0-8176-4409-1_1.

[25] A. A. Kurzhanskiy and P. Varaiya, "Ellipsoidal techniques for reachability analysis of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 1, pp. 26–38, 2007. DOI: 10.1109/TAC.2006.887900.

[26] J.-P. Aubin, A. Bayen, and P. Saint-Pierre, "Viability theory: New directions," Jan. 2011. DOI: 10.1007/978-3-642-16684-6.

[27] J.-P. Aubin, "A survey of viability theory," *SIAM Journal on Control and Optimization*, vol. 28, no. 4, pp. 749–788, 1990. DOI: 10.1137/0328044. eprint: https://doi.org/10.1137/0328044. [Online]. Available: https://doi.org/10.1137/0328044.

[28] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," *IFAC Proceedings Volumes*, vol. 40, no. 12, pp. 462–467, 2007, 7th IFAC Symposium on Nonlinear Control Systems, ISSN: 1474-6670. DOI: https://doi.org/10.3182/20070822-3-ZA-2920.00076. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016355690.

[29] I. Weintraub, M. Pachter, and E. Garcia, "An introduction to pursuit-evasion differential games," *American Control Conference*, pp. 1049–1066, 2020.

[30] D. W. Oyler, P. T. Kabamba, and A. R. Girard, "Pursuit–evasion games in the presence of obstacles," *Automatica*, vol. 65, pp. 1–11, 2016.

[31] E. Garcia, D. W. Casbeer, A. Von Moll, and M. Pachter, "Multiple pursuer multiple evader differential games," *IEEE Transactions on Automatic Control*, pp. 1–1, 2020. DOI: 10.1109/TAC.2020.3003840.

[32] J. Fisac, M. Chen, C. Tomlin, and S. Sastry, "Reach-avoid problems with time-varying dynamics, targets and constraints," 2014, pp. 11–20. DOI: 10.1145/2728606.2728612.

[33] O. Bokanowski and H. Zidani, "Minimal time problems with moving targets and obstacles," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 2589–2593, 2011, 18th IFAC World Congress, ISSN: 1474-6670. DOI: https://doi.org/10.3182/20110828-6-IT-1002.02261. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016440036.

[34] H. Huang, J. Ding, W. Zhang, and C. J. Tomlin, "A differential game approach to planning in adversarial scenarios: A case study on capture-the-flag," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1451–1456. DOI: 10.1109/ICRA.2011.5980264.

[35] E. Garcia, D. W. Casbeer, and M. Pachter, "The capture-the-flag differential game," *IEEE Conference on Decision and Control (CDC)*, pp. 4167–4172, 2018.

[36] L. C. Evans and P. E. Souganidis, "Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations," *Indiana University Mathematics Journal*, vol. 33, no. 5, pp. 773–797, 1984, ISSN: 00222518, 19435258. [Online]. Available: http://www.jstor.org/stable/45010271.

[37] R. Takei, H. Huang, J. Ding, and C. J. Tomlin, "Time-optimal multi-stage motion planning with guaranteed collision avoidance via an open-loop game formulation," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 323–329. DOI: 10.1109/ICRA.2012.6225074.

[38] E. Garcia, D. W. Casbeer, and M. Pachter, "Optimal strategies for a class of multi-player reach-avoid differential games in 3d space," in *IEEE Robotics and Automation Letters*, vol. 5, 2020, pp. 4257–4264.

[39] R. Yan, Z. Shi, and Y. Zhong, "Task assignment for multiplayer reach–avoid games in convex domains via analytical barriers," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 107–124, 2019.

[40] R. Yan, Z. Shi, Y. Zhong, and F. Bullo, "Matching-based capture strategies for 3d heterogeneous multiplayer reach-avoid differential games," *arXiv*, 2019. DOI: 1909.11881.

[41] M. Chen, J. Fisac, S. Sastry, and C. Tomlin, "Safe sequential path planning of multi-vehicle systems via double-obstacle hamilton-jacobi-isaacs variational inequality," Jul. 2015, pp. 3304–3309. DOI: 10.1109/ECC.2015.7331044.

[42] M. Chen, Z. Zhou, and C. J. Tomlin, "Multiplayer reach-avoid games via pairwise outcomes," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1451–1457, Mar. 2017, ISSN: 1558-2523. DOI: 10.1109/TAC.2016.2577619.

[43] H. Huang, "Reachability-based control for human and autonomous agents in adversarial games," Ph.D. dissertation, Stanford University, Jul. 2012.

[44] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations (Systems & Control: Foundations & Applications)*. Birkhäuser, Boston, 1997.

[45] J. A. Sethian, "Fast marching methods," *SIAM Review*, vol. 41, no. 2, pp. 199–235, 1999. DOI: 10.1137/S0036144598347059. [Online]. Available: https://doi.org/10.1137/S0036144598347059.

[46] G. Peyre. (2009). "Toolbox fast marching." https://www.mathworks.com/matlabcentral/fileexchange/6110-toolbox-fast-marching.

[47] E. Masehian and N. Mohamadnejad, "Path planning of nonholonomic flying robots using a new virtual obstacle method," in *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, Oct. 2015, pp. 612–617. DOI: 10.1109/ICRoM.2015.7367853.

[48] Z. Zhou, R. Takei, H. Huang, and C. J. Tomlin, "A general, open-loop formulation for reach-avoid games," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, Dec. 2012, pp. 6501–6506. DOI: 10.1109/CDC.2012.6426643.

[49] E. Garcia, D. W. Casbeer, and M. Pachter, "Optimal strategies of the differential game in a circular region," *IEEE Control Systems Letters*, vol. 4, no. 2, pp. 492–497, 2020.

[50] S. Zhao, D. V. Dimarogonas, Z. Sun, and D. Bauso, "A general approach to coordination control of mobile agents with motion constraints," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1509–1516, 2018. DOI: 10.1109/TAC.2017.2750924.

[51] L. Ma, F. He, and Y. Yao, "A 2-d coverage-based guidance algorithm for the multi-stage cooperative interception," in *2019 Chinese Control Conference (CCC)*, 2019, pp. 4224–4229. DOI: 10.23919/ChiCC.2019.8865550.

[52] X. Zeng, S. Liang, and Y. Hong, "Distributed variational equilibrium seeking of multi-coalition game via variational inequality approach," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 940–945, 2017, 20th IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.120. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896317301556.

[53] X. R. Li and Y. Bar-Shalom, "Tracking in clutter with nearest neighbor filters: Analysis and performance," *IEEE transactions on aerospace and electronic systems*, vol. 32, no. 3, pp. 995–1010, 1996.

[54] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, IEEE, 1980, pp. 807–812.

[55] K.-C. Chang and Y. Bar-Shalom, "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements and maneuvers," *IEEE Transactions on Automatic control*, vol. 29, no. 7, pp. 585–594, 1984.

[56] R. J. Fitzgerald, "Development of practical pda logic for multitarget tracking by microprocessor," in *1986 American Control Conference*, IEEE, 1986, pp. 889–898.

[57] D. Reid, "An algorithm for tracking multiple targets," *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[58] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[59] R. P. Mahler, *Statistical multisource-multitarget information fusion*. Artech House, Inc., 2007.

[60] R. P. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[61] B.-N. Vo and W.-K. Ma, "A closed-form solution for the probability hypothesis density filter," in *2005 7th International Conference on Information Fusion*, IEEE, vol. 2, 2005, 8–pp.

[62] B.-N. Vo and W.-K. Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[63] M. Yazdian-Dehkordi, Z. Azimifar, and M. A. Masnadi-Shirazi, "Penalized gaussian mixture probability hypothesis density filter for multiple target tracking," *Signal Processing*, vol. 92, no. 5, pp. 1230–1242, 2012.

[64] B. Han, C. Paulson, T. Lu, D. Wu, and J. Li, "Tracking of multiple objects under partial occlusion," in *Automatic Target Recognition XIX*, International Society for Optics and Photonics, vol. 7335, 2009, p. 733 515.

[65] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 1815–1821.

[66] Y. Chou, H. Yoon, and S. Sankaranarayanan, "Predictive runtime monitoring of vehicle models using bayesian estimation and reachability analysis," in *Intl. Conference on Intelligent Robots and Systems (IROS)*, 2020.

[67] S. Oh, I. Hwang, and S. Sastry, "Distributed multitarget tracking and identity management," *Journal of guidance, control, and dynamics*, vol. 31, no. 1, pp. 12–29, 2008.

[68] I. Hwang, H. Balakrishnan, K. Roy, and C. Tomlin, "Multiple-target tracking and identity management in clutter, with application to aircraft tracking," in *Proceedings of the 2004 American Control Conference*, IEEE, vol. 4, 2004, pp. 3422–3428.

[69] M. Nagumo, "Über die lage der integralkurven gewöhnlicher differentialgleichungen," *Proceedings of the Physico-Mathematical Society of Japan*, vol. 24, pp. 551–559, 1942.

[70] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," R. Alur and G. J. Pappas, Eds., pp. 477–492, 2004.

[71] S. Prajna, "Barrier certificates for nonlinear model validation," vol. 3, 2884–2889 Vol.3, 2003. DOI: 10.1109/CDC.2003.1273063.

[72] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," pp. 6271–6278, 2014. DOI: 10.1109/CDC.2014.7040372.

[73] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017. DOI: 10.1109/TAC.2016.2638961.

[74] S.-C. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," pp. 4542–4548, 2015. [Online]. Available: http://ames.caltech.edu/ACC_2015_CBF_final.pdf.

[75] Q. Nguyen, A. Hereid, J. W. Grizzle, A. D. Ames, and K. Sreenath, "3d dynamic walking on stepping stones with control barrier functions," pp. 827–834, 2016. DOI: 10.1109/CDC.2016.7798370.

[76] Q. Nguyen and K. Sreenath, "Safety-critical control for dynamical bipedal walking with precise footstep placement," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 147–154, 2015, Analysis and Design of Hybrid Systems ADHS, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.11.167. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315024258.

[77] X. Xu, J. W. Grizzle, P. Tabuada, and A. D. Ames, "Correctness guarantees for the composition of lane keeping and adaptive cruise control," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1216–1229, 2018. DOI: 10.1109/TASE.2017.2760863.

[78] X. Xu, T. Waters, D. Pickem, P. Glotfelter, M. Egerstedt, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Realizing simultaneous lane keeping and adaptive speed regulation on accessible mobile robot testbeds," pp. 1769–1775, 2017. DOI: 10.1109/CCTA.2017.8062713.

[79] G. Wu and K. Sreenath, "Safety-critical control of a planar quadrotor," pp. 2252–2258, 2016. DOI: 10.1109/ACC.2016.7525253.

[80] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," pp. 3293–3298, 2017. DOI: 10.1109/ICRA.2017.7989375.

[81] T. Gurriet, A. Singletary, J. Reher, L. Ciarletta, E. Feron, and A. Ames, "Towards a framework for realizable safety critical control through active set invariance," pp. 98–106, 2018. DOI: 10.1109/ICCPS.2018.00018.

[82] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 68–73, 2015, Analysis and Design of Hybrid Systems ADHS, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.11.154. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S240589631502412X.

[83] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017. DOI: 10.1109/TRO.2017.2659727.

[84] B. Capelli and L. Sabattini, "Connectivity maintenance: Global and optimized approach through control barrier functions," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 5590–5596. DOI: 10.1109/ICRA40945.2020.9197109.

[85] M. Egerstedt, J. N. Pauli, G. Notomista, and S. Hutchinson, "Robot ecology: Constraint-based control design for long duration autonomy," *Annual Reviews in Control*, vol. 46, pp. 1–7, 2018, ISSN: 1367-5788. DOI: https://doi.org/10.1016/j.arcontrol.2018.09.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S136757881830141X.

[86] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 2460–2465.

[87] W. Jin, Z. Wang, Z. Yang, and S. Mou, "Neural Certificates for Safe Control Policies," *arXiv e-prints*, arXiv:2006.08465, arXiv:2006.08465, Jun. 2020. arXiv: 2006.08465 [eess.SY].

[88] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *2020 59th IEEE Conference on Decision and Control (CDC)*, IEEE, 2020, pp. 3717–3724.

[89] M. Saveriano and D. Lee, "Learning barrier functions for constrained motion planning with dynamical systems," *arXiv preprint arXiv:2003.11500*, 2020.

[90] M. Srinivasan, A. Dabholkar, S. Coogan, and P. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," *arXiv preprint arXiv:2003.04950*, 2020.

[91] E. Squires, P. Pierpaoli, and M. Egerstedt, "Constructive barrier certificates with applications to fixed-wing aircraft collision avoidance," in *2018 IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, 2018, pp. 1656–1661.

[92] A. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," Mar. 2019. DOI: 10.23919/ECC.2019.8796030.

[93] P. Jagtap, G. Pappas, and M. Zamani, "Control barrier functions for unknown nonlinear systems using gaussian processes *," *59th IEEE Conference on Decision and Control (CDC)*, pp. 3699–3704, Dec. 2020. DOI: 10.1109/CDC42340.2020.9303847.